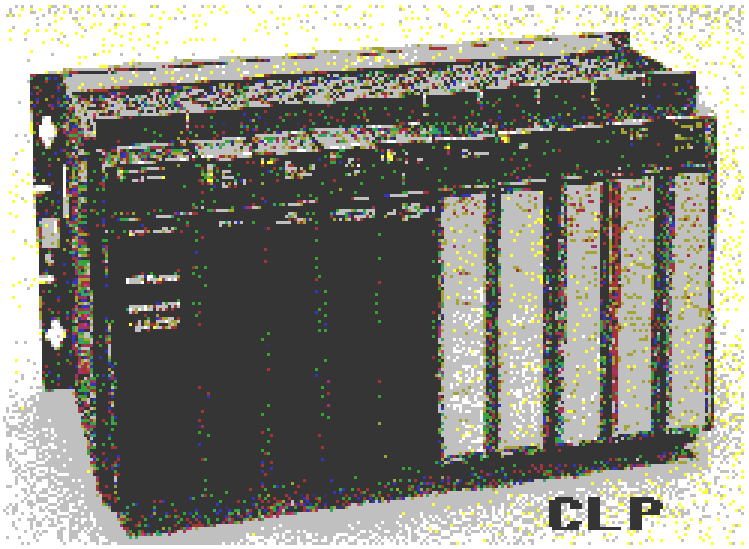
CONTROLADOR LÓGICO PROGRAMÁVEL



ÍNDICE DE TÓPICOS

I . Introdução

II. Informações Gerais
- II.1. DESCRIÇÃO
- II.2. CARACTERÍSTICAS
- II.3. HISTÓRICO
- II.4. EVOLUÇÃO
- II.5. APLICAÇÕES

III. Estrutura Básica
- III.1. UNIDADE CENTRAL DE PROCESSAMENTO (UCP)
- III.2. MEMÓRIA
- III.3. DISPOSITIVOS DE ENTRADA E SAÍDA
- III.3.1. CARACTERÍSTICAS DAS ENTRADAS E SAÍDAS - E/S
  - MÓDULOS DE ENTRADA
  - TRATAMENTO DE SINAL DE ENTRADA
  - MÓDULOS DE SAÍDA
  - TRATAMENTO DE SINAL DE SAÍDA
- III.4. TERMINAL DE PROGRAMAÇÃO

IV. Princípio de Funcionamento de um CLP
- IV.1. ESTADOS DE OPERAÇÃO
- IV.2. FUNCIONAMENTO

V. Linguagem de Programação
- V.1. CLASSIFICAÇÃO
  - LINGUAGEM DE BAIXO NÍVEL
  - LINGUAGEM DE ALTO NÍVEL

VI. Programação de Controladores Programáveis
VI.1. DIAGRAMA DE CONTATOS
VI.2. DIAGRAMA DE BLOCOS LÓGICOS
VI.3. LISTA DE INSTRUÇÃO
VI.4. LINGUAGEM CORRENTE
VI.5. ANÁLISE DAS LINGUAGUES DE PROGRAMAÇÃO
Quanto a Forma de Programação
Quanto a Forma de Representação
Documentação
Conjunto de Instruções
VI.6. NORMALIZAÇÃO

VII. Programação em Ladder
VII.1. DESENVOLVIMENTO DO PROGRAMA LADDER
VII.1.1 ASSOCIAÇÃO DE CONTATOS NO LADDER.
VII.1.2. INSTRUÇÕES
VII.1.3. INSTRUÇÕES BÁSICAS
INSTRUÇÃO DE TEMPORIZAÇÃO
INSTRUÇÃO DE CONTAGEM
INSTRUÇÃO MOVER
INSTRUÇÃO COMPARAR
VII.1.4. INSTRUÇÕES MATEMÁTICAS
INSTRUÇÃO SOMA
INSTRUÇÃO SUBTRAÇÃO
INSTRUÇÃO MULTIPLICAÇÃO
INSTRUÇÃO DIVISÃO
VII.1.5. INSTRUÇÕES LÓGICAS
INSTRUÇÃO AND
INSTRUÇÃO OR
INSTRUÇÃO XOR

VIII. Noções de Sistema Supervisório ï Intouch.

IX. Noções de Blocos I/O Remotos

I . Introdução

O Controlador Lógico Programável, ou simplesmente CLP, tem revolucionado os comandos e controles industriais desde seu surgimento na década de 70.

Antes do surgimento dos CLPs as tarefas de comando e controle de máquinas e processos industrias eram feitas por relés eletromagnéticos, especialmente projetados para este fim.

II. Informações Gerais

II.1. DESCRIÇÃO

O primeiro CLP surgiu na indústria automobilística, até então um usuário em potencial dos relés eletromagnéticos utilizados para controlar operações sequenciadas e repetitivas numa linha de montagem. A primeira geração de CLPs utilizou componentes discretos como transistores e CIs com baixa escala de integração.

Este equipamento foi batizado nos Estados Unidos como PLC (*Programable Logic Control*), em português CLP (*Controlador Lógico Programável*) e este termo é registrado pela *Allen Bradley* (fabricante de CLPs).

Definição segundo a ABNT (Associação Brasileira de Normas Técnicas)

É um equipamento eletrônico digital com hardware e software compatíveis com aplicações industriais.

Definição segundo a Nema (National Electrical Manufacturers Association)

Aparelho eletrônico digital que utiliza uma memória programável para o armazenamento interno de instruções para implementações específicas, tais como lógica, seqüenciamento, temporização, contagem e aritmética, para controlar, através de módulos de entradas e saídas, vários tipos de máquinas ou processos.

II.2. CARACTERÍSTICAS

Basicamente, um controlador programável apresenta as seguintes características:

⇒ **Hardware e/ou dispositivo de controle de fácil e rápida programação ou reprogramação, com a mínima interrupção da produção.**
⇒ **Capacidade de operação em ambiente industrial .**
⇒ **Sinalizadores de estado e módulos tipo plug-in de fácil manutenção e substituição.**
⇒ **Hardware ocupando espaço reduzido e apresentando baixo consumo de energia.**
⇒ **Possibilidade de monitoração do estado e operação do processo ou sistema, através da comunicação com computadores.**
⇒ **Compatibilidade com diferentes tipos de sinais de entrada e saída.**
⇒ **Capacidade de alimentar, de forma contínua ou chaveada, cargas que consomem correntes de até 2 A.**
⇒ **Hardware de controle que permite a expansão dos diversos tipos de módulos, de acordo com a necessidade.**
⇒ **Custo de compra e instalação competitivo em relação aos sistemas de controle convencionais.**
⇒ **Possibilidade de expansão da capacidade de memória.**
⇒ **Conexão com outros CLPs através de rede de comunicação.**

II.3. HISTÓRICO

O controlador programável nasceu praticamente dentro da indústria automobilística americana, especificamente na Hydromic Division da General Motors, em 1968, devido a grande dificuldade de se mudar a lógica de controle de painéis de comando a cada mudança na linha de montagem. Estas mudanças implicavam em altos gastos de tempo e dinheiro.

Sob a liderança do engenheiro Richard Morley, foi preparada uma especificação que refletia os sentimentos de muitos usuários de relés, não só da indústria automobilística como de toda a indústria manufatureira.

Nascia assim a indústria de controladores programáveis, hoje com um mercado mundial estimado em 4 bilhões de dólares anuais. Que no Brasil é estimado em 50 milhões de dólares anuais.

II.4. EVOLUÇÃO

Desde o seu aparecimento até hoje, muita coisa evolui nos controladores lógicos. Esta evolução está ligada diretamente ao desenvolvimento tecnológico da informática em suas características de software e de hardware.

O que no seu surgimento era executado com componentes discretos, hoje se utiliza de microprocessadores e microcontroladores de última geração, usando técnicas de processamento paralelo, inteligência artificial, redes de comunicação, fieldbus, etc.

Até recentemente não havia nenhuma padronização entre fabricantes, apesar da maioria utilizar as mesmas normas construtivas. Porém, pelo menos ao nível de software aplicativo, os controladores programáveis podem se tornar compatíveis com a adoção da norma IEC 1131-3, que prevê a padronização da linguagem de programação e sua portabilidade.

Outra novidade que está sendo incorporada pelos controladores programáveis é o fieldbus (barramento de campo), que surge como uma proposta de padronização de sinais a nível de chão-de-fábrica. Este barramento se propõe a diminuir sensivelmente o número de condutores usados para interligar os sistemas de controle aos sensores e atuadores, além de propiciar a distribuição da inteligência por todo o processo.

Hoje os CLPs oferecem um considerável número de benefícios para aplicações industriais, que podem ressaltar em economia que excede o custo do CLP e devem ser considerados quando da seleção de um dispositivo de controle industrial. As vantagens de sua utilização, comparados a outros dispositivos de controle industrial incluem:

- ⇒ **Menor Ocupação de espaço;**
- ⇒ **Potência elétrica requerida menor;**
- ⇒ **Reutilização;**
- ⇒ **Programável, se ocorrerem mudanças de requisitos de controle;**
- ⇒ **Confiabilidade maior;**
- ⇒ **Manutenção mais fácil;**
- ⇒ **Maior flexibilidade, satisfazendo um maior número de aplicações;**
- ⇒ **Permite a interface através de rede de comunicação com outros CLPs e microcomputadores;**
- ⇒ **projeto do sistema mais rápido.**

Todas estas considerações mostram a evolução de tecnologia, tanto de hardware quanto de software, o que permite o seu acesso a um maior número de pessoas tanto nos projetos de aplicação de controladores programáveis quanto na sua programação.

II.5. APLICAÇÕES

O controlador programável existe para automatizar processos industriais, sejam de sequênciamento, intertravamento, controle de processos, batelada, etc.

Este equipamento tem seu uso tanto na área de automação da manufatura, de processos contínuos, elétrica, predial, entre outras.

Praticamente não existem ramos de aplicações industriais onde não se possa aplicar os CLPs, entre elas tem-se:

⇒ **Máquinas industriais (operatrizes, injetoras de plástico, têxteis, calçados);**
⇒ **Equipamentos industriais para processos (siderurgia, papel e celulose, petroquímica, química, alimentação, mineração, etc);**
⇒ **Equipamentos para controle de energia (demanda, fator de carga);**
⇒ **Controle de processos com realização de sinalização, intertravamento e controle PID;**
⇒ **Aquisição de dados de supervisão em: fábricas, prédios inteligentes, etc;**
⇒ **Bancadas de teste automático de componentes industriais;**
⇒ **Etc.**

Com a tendência dos CLPs terem baixo custo, muita inteligência, facilidade de uso e massificação das aplicações, a utilização deste equipamento não será apenas nos processos mas também nos produtos. Poderemos encontrá-lo em produtos eletrodomésticos, eletrônicos, residências e veículos.

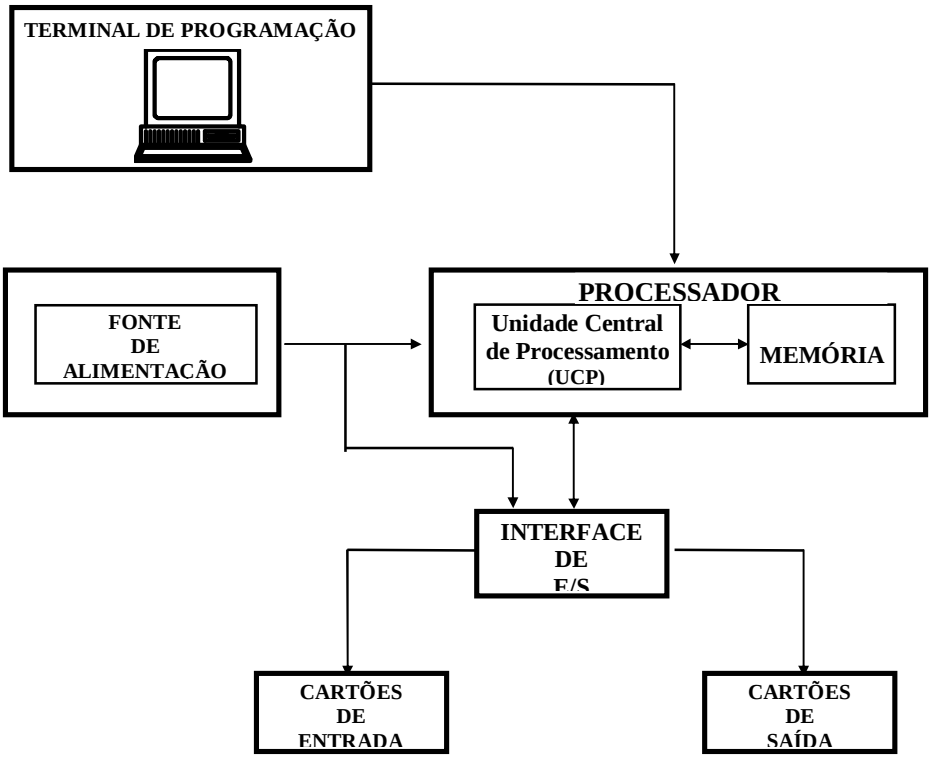
III. Estrutura Básica

O controlador programável tem sua estrutura baseada no hardware de um computador, tendo portanto uma unidade central de processamento (UCP), interfaces de entrada e saída e memórias.

As principais diferenças em relação a um computador comum estão relacionadas a qualidade da fonte de alimentação, que possui características ótimas de filtragem e estabilização, interfaces de E/S imune a ruídos e um invólucro específico para aplicações industriais.

Temos também um terminal usado para programação do CLP.

O diagrama de blocos a seguir, ilustra a estrutura básica de um controlador programável:



Dentre as partes integrantes desta estrutura temos:

⇒ **UCP**
⇒ **Memória**
⇒ **E/S (Entradas e Saídas)**
⇒ **Terminal de Programação**

III.1. UNIDADE CENTRAL DE PROCESSAMENTO (UCP)

A **U**nidade **C**entral de **P**rocessamento (UCP) é responsável pelo processamento do programa, isto é, coleta os dados dos cartões de entrada, efetua o processamento segundo o programa do usuário, armazenado na memória, e envia o sinal para os cartões de saída como resposta ao processamento.

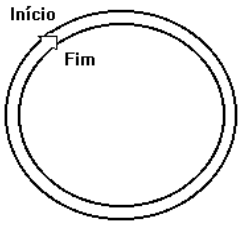
Geralmente, cada CLP tem uma UCP, que pode controlar vários pontos de E/S (entradas e saídas) fisicamente compactadas a esta unidade - é a filosofia compacta de fabricação de CLPs, ou constituir uma unidade separada, conectada a módulos onde se situam cartões de entrada e saída, - esta é a filosofia modular de fabricação de CLPs.

Este processamento poderá ter estruturas diferentes para a execução de um programa, tais como:

⇒ **Processamento cíclico;**
⇒ **Processamento por interrupção;**
⇒ **Processamento comandado por tempo;**
⇒ **Processamento por evento.**

Processamento Cíclico

É a forma mais comum de execução que predomina em todas as UCPs conhecidas, e de onde vem o conceito de varredura, ou seja, as instruções de programa contidas na memória, são lidas uma após a outra seqüencialmente do início ao fim, daí retornando ao início ciclicamente.



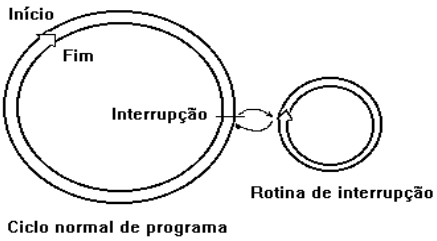
Um dado importante de uma UCP é o seu tempo de ciclo, ou seja, o tempo gasto para a execução de uma varredura. Este tempo está relacionado com o tamanho do programa do usuário (em média 10 ms a cada 1.000 instruções).

Processamento por interrupção

Certas ocorrências no processo controlado não podem, algumas vezes, aguardar o ciclo completo de execução do programa. Neste caso, ao reconhecer uma ocorrência deste tipo, a UCP interrompe o ciclo normal de programa e executa um outro programa chamado de rotina de interrupção.

Esta interrupção pode ocorrer a qualquer instante da execução do ciclo de programa. Ao finalizar esta situação o programa voltará a ser executado do ponto onde ocorreu a interrupção.

Uma interrupção pode ser necessária , por exemplo, numa situação de emergência onde procedimentos referentes a esta situação devem ser adotados.



Processamento comandado por tempo

Da mesma forma que determinadas execuções não podem ser dependentes do ciclo normal de programa, algumas devem ser executados a certos intervalos de tempo, as vezes muito curto, na ordem de 10 ms.

Este tipo de processamento também pode ser incarado como um tipo de interrupção, porém ocorre a intervalos regulares de tempo dentro do ciclo normal de programa.

Processamento por evento

Este é processado em eventos específicos, tais como no retorno de energia, falha na bateria e estouro do tempo de supervisão do ciclo da UCP.

Neste último, temos o chamado *Watch Dog Time* (WD), que normalmente ocorre como procedimento ao se detectar uma condição de estouro de tempo de ciclo da UCP, parando o processamento numa condição de falha e indicando ao operador através de sinal visual e as vezes sonoro.

III.2. MEMÓRIA

O sistema de memória é uma parte de vital importância no processador de um controlador programável, pois armazena todas as instruções assim como o os dados necessários para executá-las.

Existem diferentes tipos de sistemas de memória. A escolha de um determinado tipo depende:

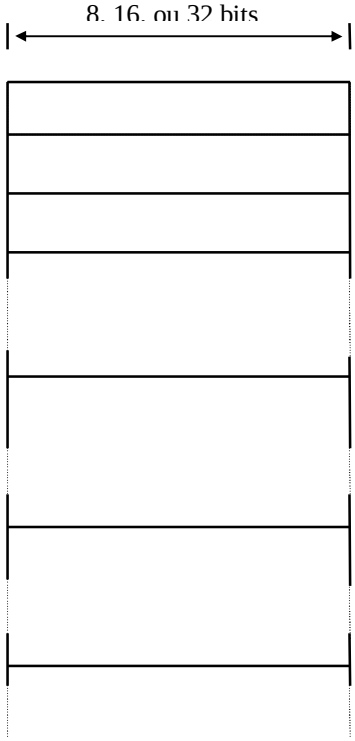
⇒ **do tipo de informação armazenada;**
⇒ **da forma como a informação será processada pela UCP.**

As informações armazenadas num sistema de memória são chamadas *palavras de memória*, que são formadas sempre com o mesmo número de bits.

A capacidade de memória de um CP é definida em função do número de palavras de memória previstas para o sistema.

Mapa de memória

A capacidade de memória de um CP pode ser representada por um mapa chamado *mapa de memória.*



ENDEREÇO DAS PALAVRAS DE MEMÓRIA

Decimal	Octal	Hexadecimal
255	377	FF
511	777	1FF
1023	1777	3FF
2047	3777	7FF
4095	7777	FFF
8191	17777	1FFF

Arquitetura de memória de um CP

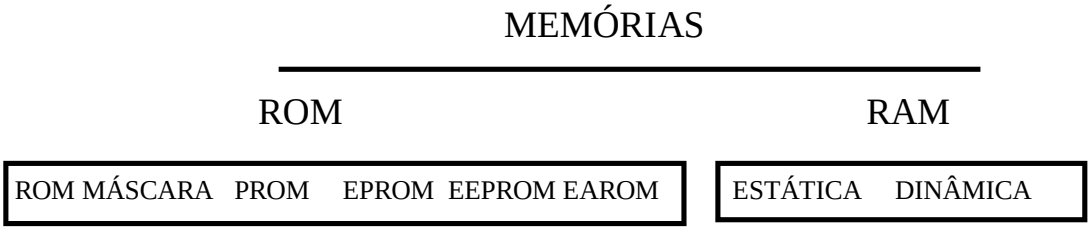
A arquitetura de memória de um controlador programável pode ser constituída por diferentes tipos de memória.

A memória do computador é onde se armazenam os dados que devem ser manipulados pelo computador (chamada memória de dados) e também onde esta armazenado o programa do computador (memória de programa).

Aparentemente não existe uma diferença física entre as memórias de programa, apenas utilizam-se memórias fixas para armazenar dados fixos ou programas e memórias que podem ser alteradas pelo sistema para armazenar dados que podem variar de acordo com o programa. Existem diversos tipos de memórias que podem ser utilizadas pelo computador: fita magnética, disco magnético e até memória de semicondutor em forma de circuito integrado.

As memórias a semicondutores podem ser divididas em dois grupos diferentes:

- Memória **ROM** (read only memory) memória apenas de leitura.
- Memória **RAM** (random acess memory) memória de acesso aleatório.



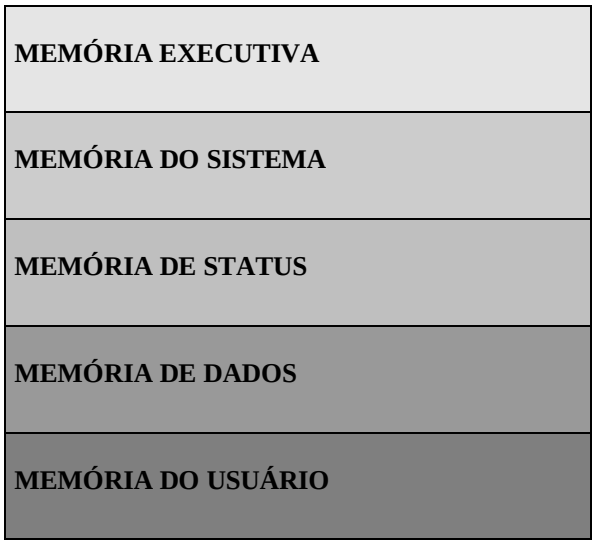
As memórias ROM são designadas como memória de programa por serem memórias que não podem ser alteradas em estado normal de funcionamento, porém têm a vantagem de não perderem as suas informações mesmo quando é desligada sua alimentação.

Tipo de Memória	*Descrição*	*Observação*
RAM DINÂMICA	Memória de acesso aleatório	- Volátil - Gravada pelo usuário - Lenta - Ocupa pouco espaço - Menor custo
RAM	Memória de acesso aleatório	- Volátil - Gravada pelo usuário - Rápida - Ocupa mais espaço - Maior custo
ROM MÁSCARA	Memória somente de leitura	- Não Volátil - Não permite apagamento - Gravada pelo fabricante
PROM	Memória programável somente de leitura	- Não volátil - Não permite apagamento - Gravada pelo usuário
EPROM	Memória programável/ apagável somente de leitura	- Não Volátil - Apagamento por ultravioleta - Gravada pelo usuário
EPROM **EEPROM** **FLASH EPROM**	Memória programável/ apagável somente de leitura	- Não Volátil - Apagável eletricamente - Gravada pelo usuário

Estrutura

Independente dos tipos de memórias utilizadas, o mapa de memória de um controlador programável pode ser dividido em cinco áreas principais:

- ⇒ **Memória executiva**
- ⇒ **Memória do sistema**
- ⇒ **Memória de status dos cartões de E/S ou Imagem**
- ⇒ **Memória de dados**
- ⇒ **Memória do usuário**



Memória Executiva

É formada por memórias do tipo **ROM** ou **PROM** e em seu conteúdo está armazenado o *sistema operacional* responsável por todas as operações que são realizadas no CLP.

O usuário não tem acesso a esta área de memória.

Memória do Sistema

Esta área é formada por memórias tipo **RAM**, pois terá o seu conteúdo constantemente alterado pelo sistema operacional.

Armazena resultados e/ou operações intermediárias, geradas pelo sistema, quando necessário. Pode ser considerada como um tipo de *rascunho*.

Não pode ser acessada nem alterada pelo usuário.

Memória de Status de E/S ou Memória Imagem

A memória de status dos módulos de E/S são do tipo **RAM**. A UCP, após ter efetuado a leitura dos estados de todas as entradas, armazena essas informações na área denominada *status das entradas* ou *imagem das entradas*. Após o processamento dessas informações, os resultados serão armazenados na área denominada *status das saídas* ou *imagem das saídas*.

Memória de Dados

As memórias de dados são do tipo **RAM**, e armazenam valores do processamento das instruções utilizadas pelo programa do usuário.

Funções de temporização, contagem, artiméticas e especiais, necessitam de uma área de memória para armazenamento de dados, como:

⇒ **valores pré-selecioandos ou acumulados de contagem e temporização;**
⇒ **resultados ou variáveis de operações aritméticas;**
⇒ **resultados ou dados diversificados a serem utilizados por funções de manipulação de dados.**

Memória do Usuário

A UCP efetuará a leitura das instruções contidas nesta área a fim de executar o programa do usuário, de acordo com os procedimentos predeterminados pelo sistema operacional.

As memórias destinadas ao usuário podem ser do tipo:

⇒ **RAM**
⇒ **RAM/EPROM**
⇒ **RAM/EEPROM**

Tipo de Memória	*Descrição*
RAM	A maioria do CLPs utiliza memórias RAM para armazenar o programa d usuário assim como os dados internos do sistema. Geralmente associada a baterias internas que evitarão a perda das informações em caso de queda da alimentação.
RAM/EPROM	O usuário desenvolve o programa e efetua testes em RAM. Uma vez checado o programa, este é transferido para EPROM.
RAM/EEPROM	Esta configuração de memória do usuário permite que, uma vez definido o programa, este seja copiado em EEPROM. Uma vez efetuada a cópia, o CLP poderá operar tanto em RAM como em EEPROM. Para qualquer modificação bastará um comando via software, e este tipo de memória será apagada e gravada eletricamente.

III.3. DISPOSITIVOS DE ENTRADA E SAÍDA

Os dispositivos de entrada e saída são os circuitos responsáveis pela interação entre o homem e a máquina; são os dispositivos por onde o homem pode introduzir informações na máquina ou por onde a máquina pode enviar informações ao homem. Como dispositivos de entrada podemos citar os seguintes exemplos: leitor de fitas magnéticas, leitor de disco magnético, leitor de cartão perfurado, leitor de fita perfurada, teclado, painel de chaves, conversor A/D, mouse, scaner, etc. Estes dispositivos tem por função a transformação de dados em sinais elétricos codificados para a unidade central de processamento.

Como dispositivos de saída podemos citar os seguintes exemplos: gravador de fitas magnéticas, gravador de discos magnéticos, perfurador de cartão, perfurador de fita, impressora, vídeo, display, conversor D/A, canal de som, etc. Todos eles tem por função a transformação de sinais elétricos codificados pela máquina em dados que possam ser manipulados posteriormente ou dados que são imediatamente entendidos pelo homem.

Estes dispositivos são conectados à unidade central de processamento por intermédio de "portas" que são interfaces de comunicação dos dispositivos de entrada e saída.

A estrutura de E/S (entradas e saídas) é encarregada de filtrar os vários sinais recebidos ou enviados para os componentes externos do sistema de controle. Estes componentes ou dispositivos no campo podem ser botões, chaves de fim de curso, contatos de relés, sensores analógicos, termopares, chaves de seleção, sensores indutivos, lâmpadas sinalizadoras, display de LEDs, bobinas de válvulas direcionais elétricas, bobinas de relés, bobinas de contatoras de motores, etc.

Em ambientes industriais, estes sinais de E/S podem conter ruído elétrico, que pode causar operação falha da UCP se o ruído alcançar seus circuitos. Desta forma, a estrutura de E/S protege a UCP deste tipo de ruído, assegurando informações confiáveis. A fonte de alimentação das E/S pode também constituir-se de uma única unidade ou de uma série de fontes, que podem estar localizadas no próprio compartimento de E/S ou constituir uma unidade à parte.

Os dispositivos do campo são normalmente selecionados, fornecidos e instalados pelo usuário final do sistema do CLP. Assim, o tipo de E/S é determinado, geralmente, pelo nível de tensão (e corrente, nas saídas) destes dispositivos. Os circuitos de E/S são tipicamente fornecidas pelos fabricantes de CLPs em módulos, cada um com 4, 8, 16 ou mais circuitos.

Além disso, a alimentação para estes dispositivos no campo deve ser fornecida externamente ao CLP, uma vez que a fonte de alimentação do CLPs é projetada para operar somente com a parte interna da estrutura de E/S e não dispositivos externos.

III.3.1. CARACTERÍSTICAS DAS ENTRADAS E SAÍDAS - E/S

A saída digital basicamente pode ser de quatro tipos: transistor, triac, contato seco e TTL podendo ser escolhido um ou mais tipos. A entrada digital pode se apresentar de várias formas, dependendo da especificação do cliente, contato seco, 24 VCC, 110 VCA, 220 VCA, etc.

A saída e a entrada analógicas podem se apresentar em forma de corrente (4 a 20 mA, 0 a 10 mA, 0 a 50 mA), ou tensão (1 a 5 Vcc, 0 a 10 VCC, -10 a 10 VCC etc). Em alguns casos é possível alterar o ranger da através de software.

MÓDULOS DE ENTRADA

Os módulos de entrada são interfaces entre os *sensores* localizados no campo e a lógica de controle de um controlador programável.

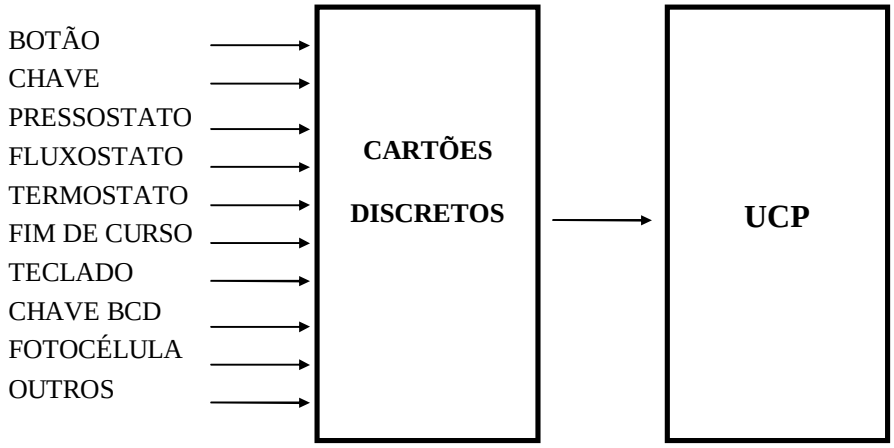
Estes módulos são constituídos de cartões eletrônicos, cada qual com capacidade para receber em certo número de variáveis.

Pode ser encontrado uma variedade muito grande de tipos de cartões, para atender as mais variadas aplicações nos ambientes industriais. Mas apesar desta grande variedade, os elementos que informam a condição de grandeza aos cartões, são do tipo:

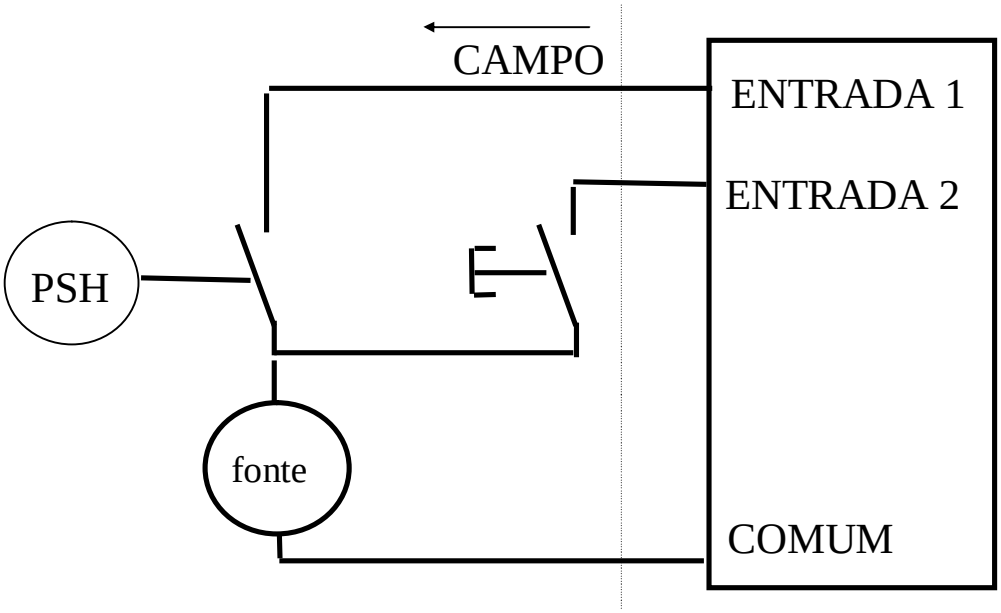
ELEMENTO DISCRETO : Trabalha com dois níveis definidos;

ELEMENTO ANALÓGICO : Trabalha dentro de uma faixa de valores.

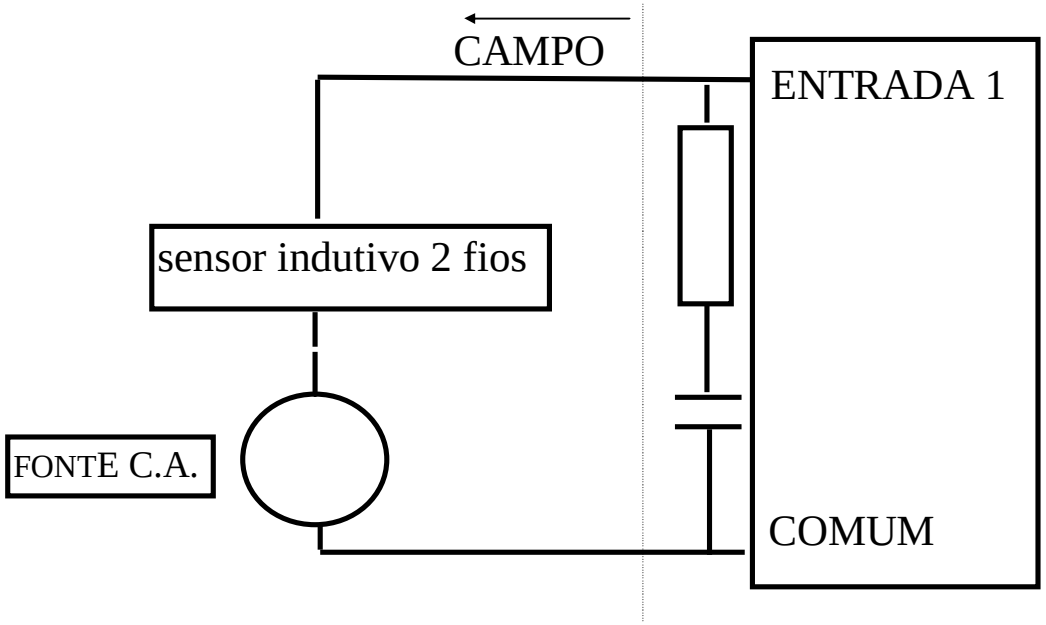
ELEMENTOS DISCRETOS



A entrada digital com fonte externa é o tipo mais utilizado, também neste caso a característica da fonte de alimentação externa dependerá da especificação do módulo de entrada. Observe que as chaves que acionam as entradas situam-se no campo.



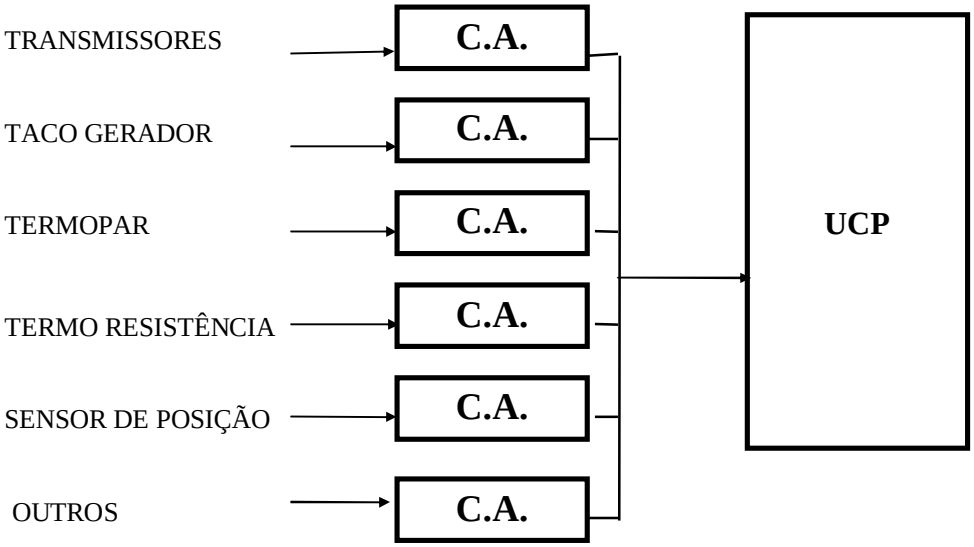
As entradas dos CLPs têm alta impedância e por isso não podem ser acionadas diretamente por um triac, como é o caso do acionamento por sensores a dois fios para CA, em razão disso é necessário, quando da utilização deste tipo de dispositivo de campo, o acréscimo de uma derivação para a corrente de manutenção do tiristor. Essa derivação consta de um circuito resistivo-capacitivo em paralelo com a entrada acionada pelo triac, cujos valores podem ser encontrados nos manuais do CLP, como visto abaixo.



Se for ser utilizado um sensor capacitivo, indutivo, óptico ou indutivo magnético, saída à transistor com alimentação de 8 a 30 VCC, basta especificar um cartão de entrada 24 VCC comum negativo ou positivo dependendo do tipo de sensor, e a saída do sensor será ligada diretamente na entrada digital do CLP.

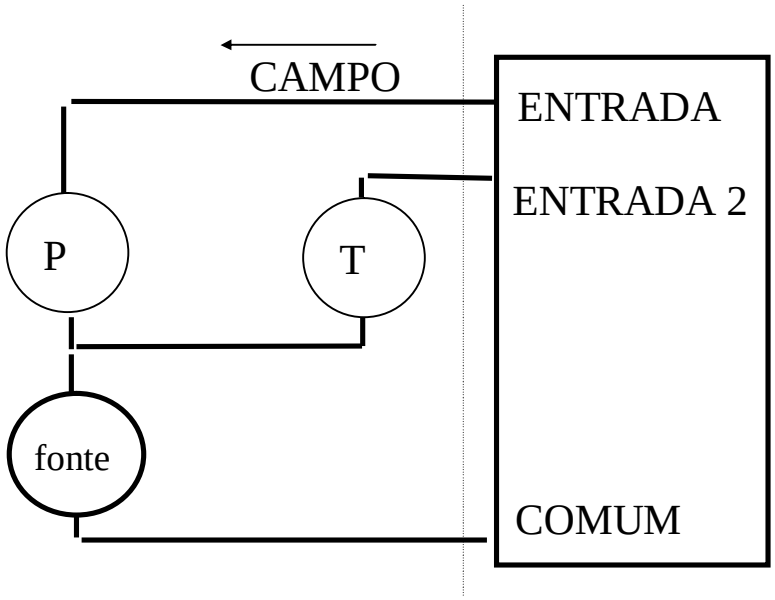
A entrada digital do tipo contato seco fica limitada aos dispositivos que apresentam como saída a abertura ou fechamento de um contato. É bom lembrar que em alguns casos uma saída do sensor do tipo transistor também pode ser usada, esta informação consta no manual de ligação dos módulos de entrada.

ELEMENTOS ANALÓGICOS

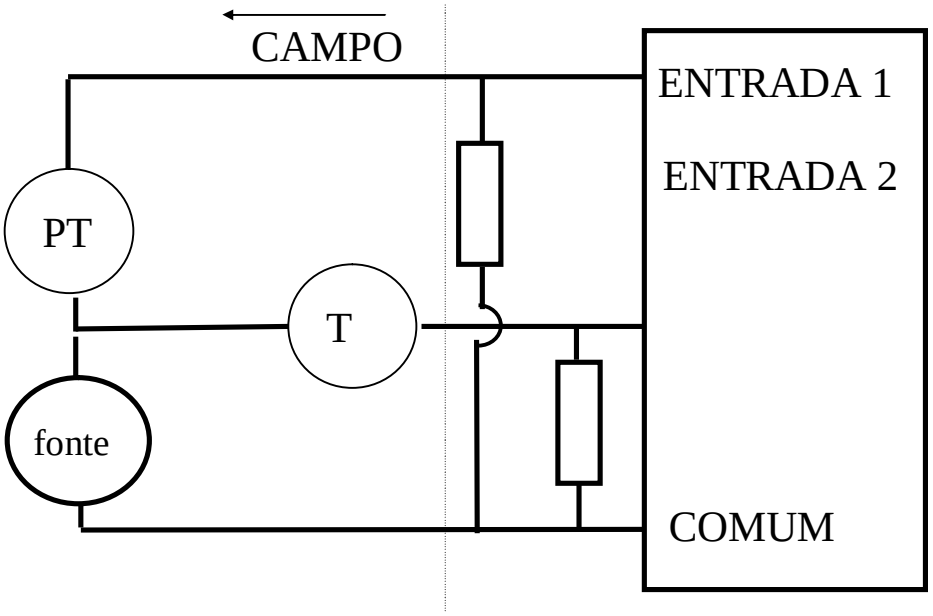


C.A. - Cartão Analógico

A entrada analógica em corrente é implementada diretamente no transmissor como mostra o diagrama.



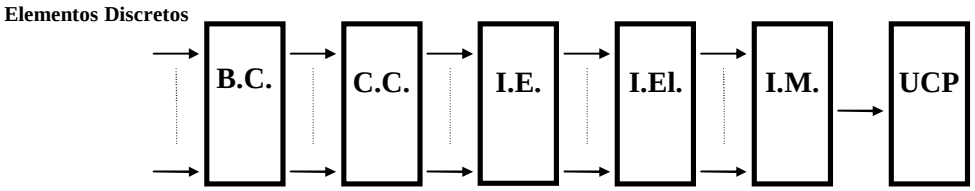
A entrada analógica em tensão necessita de um shunt para a conversão do valor de corrente em tensão, como mostra o diagrama O valor do resistor shunt dependerá da faixa de saída do transmissor e da faixa de entrada do ponto analógico. Para tal cálculo utiliza-se a lei de ohm (R = V / I).



TRATAMENTO DE SINAL DE ENTRADA

O tratamento que deve sofrer um sinal de entrada, varia em função de sua natureza, isto é, um cartão do tipo digital que recebe sinal alternado, se difere do tratamento de um cartão digital que recebe sinal contínuo e assim nos demais tipos de sinais.

A seguir é mostrado um diagrama onde estão colocados os principais componentes de um cartão de entrada digital de tensão alternada :



B.C. - Bornes de conexão: Permite a interligação entre o sensor e o cartão, geralmente se utiliza sistema ñplug-inò.

C.C. - Conversor e Condicionador : Converte em DC o sinal AC, e rebaixa o nível de tensão até atingir valores compatíveis com o restante do circuito.

I.E. - Indicador de Estado : Proporcionar indicação visual do estado funcional das entradas.

I.El. - Isolação Elétrica : Proporcionar isolação elétrica entre os sinais vindos e que serão entregues ao processador.

I.M. - Interface/Multiplexação : Informar ao processador o estado de cada variável de entrada.

MÓDULOS DE SAÍDA

Os módulos de saída são elementos que fazem a interface entre o processador e os elementos *atuadores*.

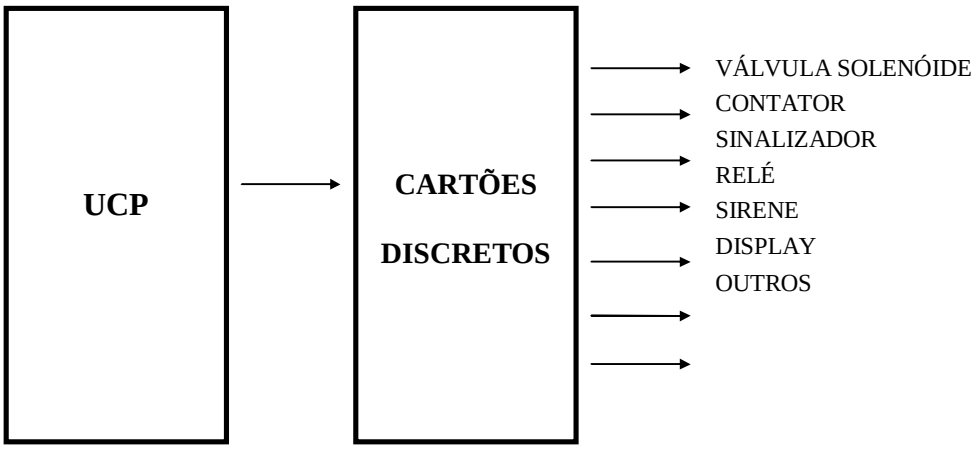
Estes módulos são constituídos de cartões eletrônicos, com capacidade de enviar sinal para os atuadores, resultante do processamento da lógica de controle.

Os cartões de saída irão atuar basicamente dois tipos:

ATUADORES DISCRETOS : Pode assumir dois estados definidos.

ATUADORES ANALÓGICOS : Trabalha dentro de uma faixa de valores.

ATUADORES DISCRETOS



De acordo com o tipo de elemento de comando da corrente das saídas, estas apresentam características que as diferem como as seguintes:

- saída a TRANSÍSTOR promove comutações mais velozes mas só comporta cargas de tensão contínua;
- saída a TRIAC tem maior vida útil que o tipo a contato seco mas só pode acionar cargas de tensão alternada;
- saída a CONTATO SECO pode acionar cargas alimentadas por tensão tanto contínua quanto alternada.

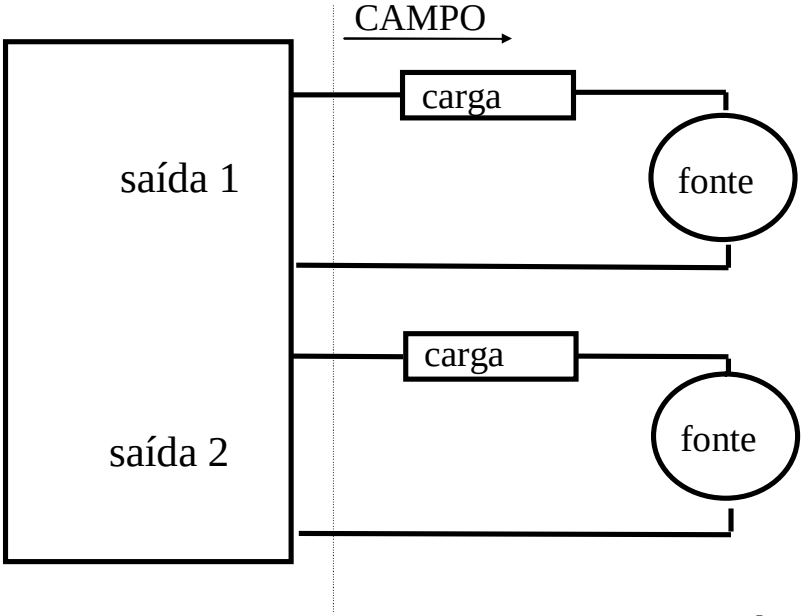
A ligação dos circuitos de entrada e ou saída é relativamente simples, dependendo apenas do tipo em questão.

A seguir vêm-se os diagramas de ligação dos vários tipos.

Uma boa prática de todo o profissional é ler o manual de instalação dos equipamentos. No que diz respeito às saídas digitais dos CLPs devem ser rigorosamente respeitados os limites de tensão, corrente e polaridade quando for o caso.

As saídas digitais independentes possuem a vantagem de poder acionar no mesmo módulo cargas de diferentes fontes sem o risco de interligá-las. Apresentam a desvantagem de consumir mais cabos.

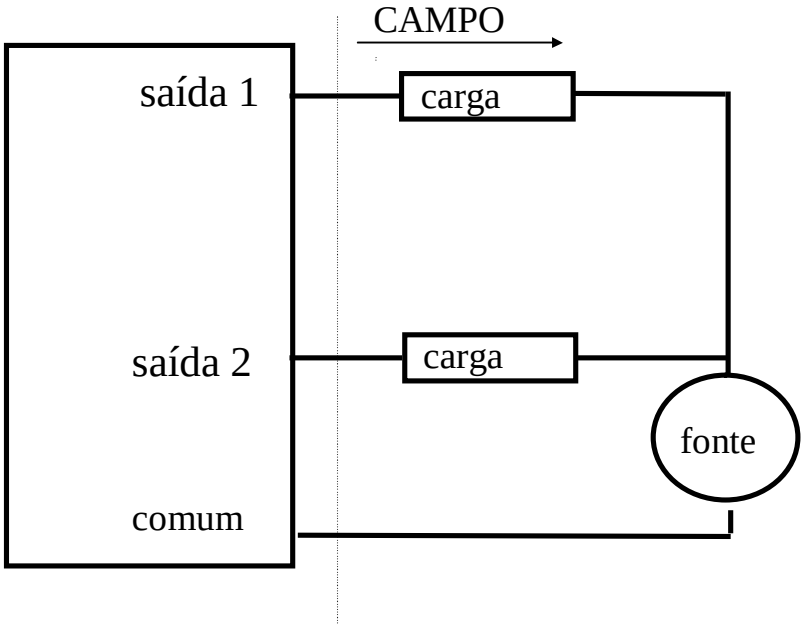
SAÍDAS DIGITAIS INDEPENDENTES



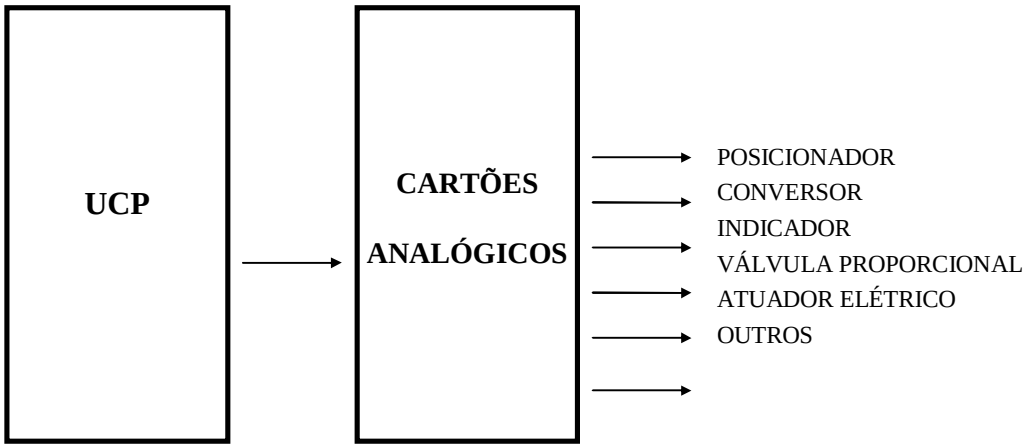
As saídas digitais com ponto comum possuem a vantagem de economia de cabo.

Se neste tipo de saída for necessário acionar cargas com fontes incompatíveis entre si, será necessária a utilização de relés cujas bobinas se energizem com as saídas do CLP e cujos contatos comandem tais cargas.

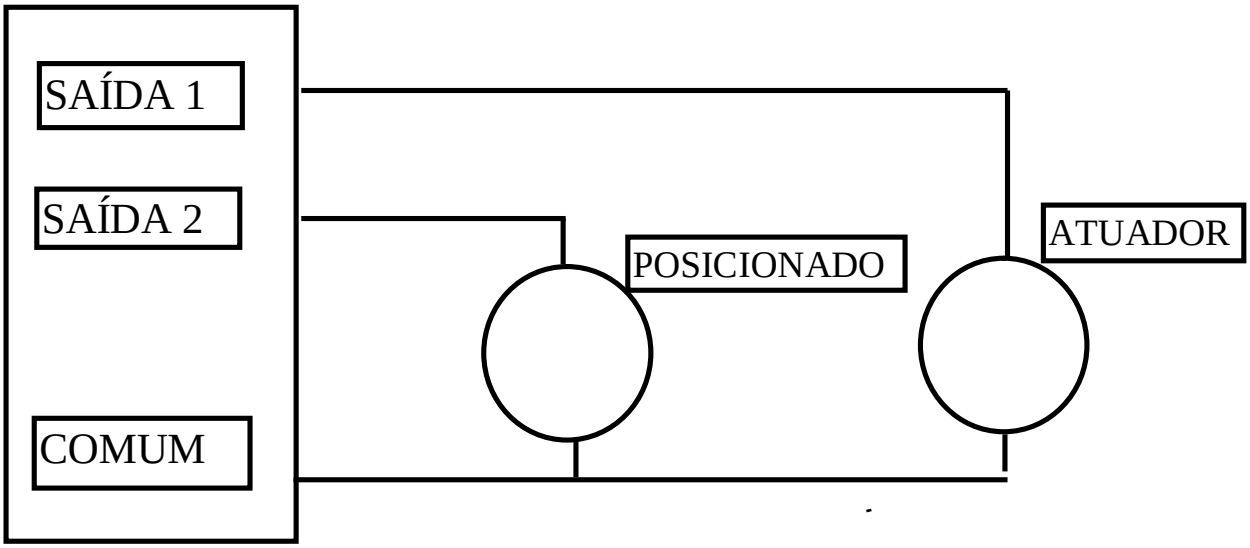
SAÍDAS DIGITAIS COM PONTO COMUM



ATUADORES ANALÓGICOS



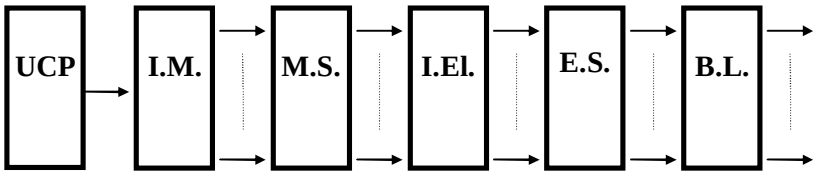
A saída analógica em corrente ou tensão é implementada diretamente no dispositivo em questão. É bom lembrar a questão da compatibilidade dos sinais, saída em tensão só pode ser ligada no dispositivo que recebe tensão e saída em corrente pode ser ligada em dispositivo que recebe corrente ou tensão, dependendo da utilização ou não do shunt de saída.



TRATAMENTO DE SINAL DE SAÍDA

Existem vários tipos de cartões de saída que se adaptam à grande variedade de atuadores existentes. Por este motivo, o sinal de saída gerado de acordo com a lógica de controle, deve ser condicionado para atender o tipo da grandeza que acionará o atuador.

A seguir é mostrado um diagrama onde estão colocados os principais componentes de um cartão de saída digital de corrente contínua :



I.M. - Interface/Multiplexação : Interpreta os sinais vindos da UCP através do barramento de dados, para os pontos de saída, correspondente a cada cartão.

M.S. - Memorizador de Sinal : Armazena os sinais que já foram multiplexados pelo bloco anterior.

I.E. - Isolação Elétrica : Proporciona isolação elétrica entre os sinais vindos do processador e os dispositivos de campo.

E.S. - Estágio de Saída : Transforma os sinais lógicos de baixa potência, em sinais capazes de operar os diversos tipos de dispositivos de campo.

B.L. - Bornes de Ligação : Permite a ligação entre o cartão e o elemento atuador, e utiliza também o sistema ñplug-inò.

III.4. TERMINAL DE PROGRAMAÇÃO

O *terminal de programação* é um dispositivo (periférico) que conectado temporariamente ao CLP, permite introduzir o programa do usuário e configuração do sistema. Pode ser um equipamento dedicado, ou seja, um terminal que só tem utilidade como programador de um determinado fabricante de CLP, ou um software que transforma um computador pessoal em um programador.

Neste periférico, através de uma linguagem, na maioria das vezes, de fácil entendimento e utilização, será feita a codificação das informações vindas do usuário numa linguagem que possa ser entendida pelo processador de um CLP. Dependendo do tipo de Terminal de Programação (TP), poderão ser realizadas funções como:

⇒ **Elaboração do programa do usuário;**
⇒ **Análise do conteúdo dos endereços de memória;**
⇒ **Introdução de novas instruções;**
⇒ **Modificação de instruções já existentes;**
⇒ **Monitoração do programa do usuário;**
⇒ **Cópia do programa do usuário em disco ou impressora.**

Os terminais de programação podem ser classificados em três tipos:

* **Terminal Dedicado Portátil;**
* **Terminal Dedicado TRC;**
* **Terminal não Dedicado;**

TERMINAL PORTÁTIL DEDICADO

Os terminais de programação portáteis, geralmente são compostos por teclas que são utilizadas para introduzir o programa do usuário. Os dados e instruções são apresentados num display que fornece sua indicação, assim como a posição da memória endereçada.

A maioria dos programadores portáteis são conectados diretamente ao CP através de uma interface de comunicação (serial). Pode-se utilizar da fonte interna do CP ou possuir alimentação própria através de bateria.

Com o advento dos computadores pessoais portáteis (Lap-Top), estes terminais estão perdendo sua função, já que pode-se executar todas as funções de programação em ambiente mais amigável, com todas as vantagens de equipamento portátil.

TERMINAL DEDICADO TRC

No caso do *Terminal de programação dedicado* tem-se como grandes desvantagens seu custo elevado e sua baixa taxa de utilização, já que sua maior utilização se dá na fase de projeto e implantação da lógica de controle.

Estes terminais são compostos por um teclado, para introdução de dados/instruções e um monitor (TRC - tubos de raios catódicos) que tem a função de apresentar as informações e condições do processo a ser controlado.

Como no caso dos terminais portáteis, com o advento da utilização de computadores pessoais, este tipo de terminal está caindo em desuso.

TERMINAL NÃO DEDICADO - PC

A utilização de um computador pessoal (PC) como terminal de programação é possível através da utilização de um software aplicativo dedicado a esta função.

Neste tipo de terminal, tem-se a vantagem da utilização de um micro de uso geral realizando o papel do programador do CLP. O custo deste hardware (PC) e software são bem menores do que um terminal dedicado além da grande vantagem de ter, após o período de implantação e eventuais manutenções, o PC disponível para outras aplicações comuns a um computador pessoal.

Outra grande vantagem é a utilização de softwares cada vez mais interativos com o usuário, utilizando todo o potencial e recursos de software e hardware disponíveis neste tipo de computador.

IV. Princípio de Funcionamento de um CLP

Um *controlador lógico programável*, tem seu funcionamento baseado num sistema de microcomputador onde se tem uma estrutura de software que realiza continuamente ciclos de varredura.

IV.1. ESTADOS DE OPERAÇÃO

Basicamente a UCP de um controlador programável possui dois estados de operação :

- **Programação**

- **Execução**

A UCP pode assumir também o estado de erro, que aponta falhas de operação e execução do programa.

Programação

Neste estado o CP *não executa programa*, isto é, não assume nenhuma lógica de controle, ficando preparado para ser configurado ou receber novos programas ou até modificações de programas já instalados. Este tipo de programação é chamada *off-line* (fora de linha).

Execução

Estado em que o CP assume a função de execução do programa do usuário. Neste estado, alguns controladores, podem sofrer modificações modificações de programa. Este tipo de programação é chamada *on-line* (em linha).

IV.2. FUNCIONAMENTO

Ao ser energizado, estando o CP no *estado de execução*, o mesmo cumpre uma rotina de inicialização gravada em seu sistema operacional. Esta rotina realiza as seguintes tarefas :

- Limpeza da memória imagem, para operandos não retentivos;

- Teste de memória RAM;

- Teste de executabilidade do programa.

Após a execução desta rotina, a UCP passa a fazer uma varredura (ciclo) constante, isto é, uma leitura seqüencial das instruções em *loop* (laço).

Entrando no *loop*, o primeiro passo a ser executado é a leitura dos pontos de entrada. Com a leitura do último ponto, irá ocorrer, a transferência de todos os valores para a chamada memória ou tabela *imagem das entradas*.

Após a gravação dos valores na tabela imagem, o processador inicia a execução do programa do usuário de acordo com as instruções armazenadas na memória.

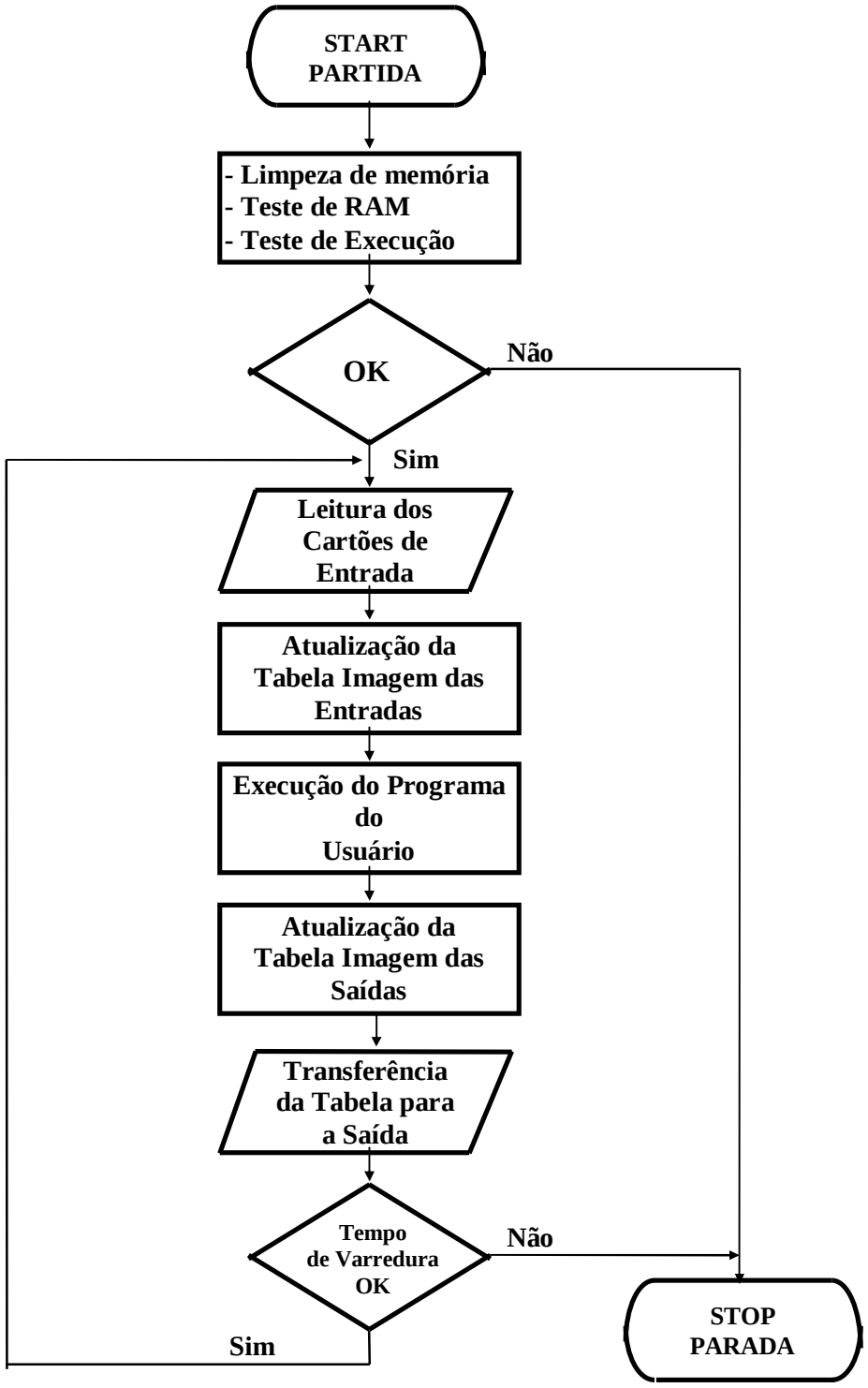
Terminando o processamento do programa, os valores obtidos neste processamento, serão transferidos para a chamada memória ou tabela *imagem das saídas*, como também a transferência de valores de outros operandos, como resultados aritméticos, contagens, etc.

Ao término da atualização da tabela imagem, será feita a transferência dos valores da tabela imagem das saídas, para os cartões de saída, fechando o *loop*. Neste momento é iniciado um novo *loop*.

Para a verificação do funcionamento da UCP, é estipulado um tempo de processamento, cabendo a um circuito chamado de *Watch Dog Time* supervisioná-lo. Ocorrendo a ultrapassagem deste tempo máximo, o funcionamento da UCP será interrompido, sendo assumido um estado de erro.

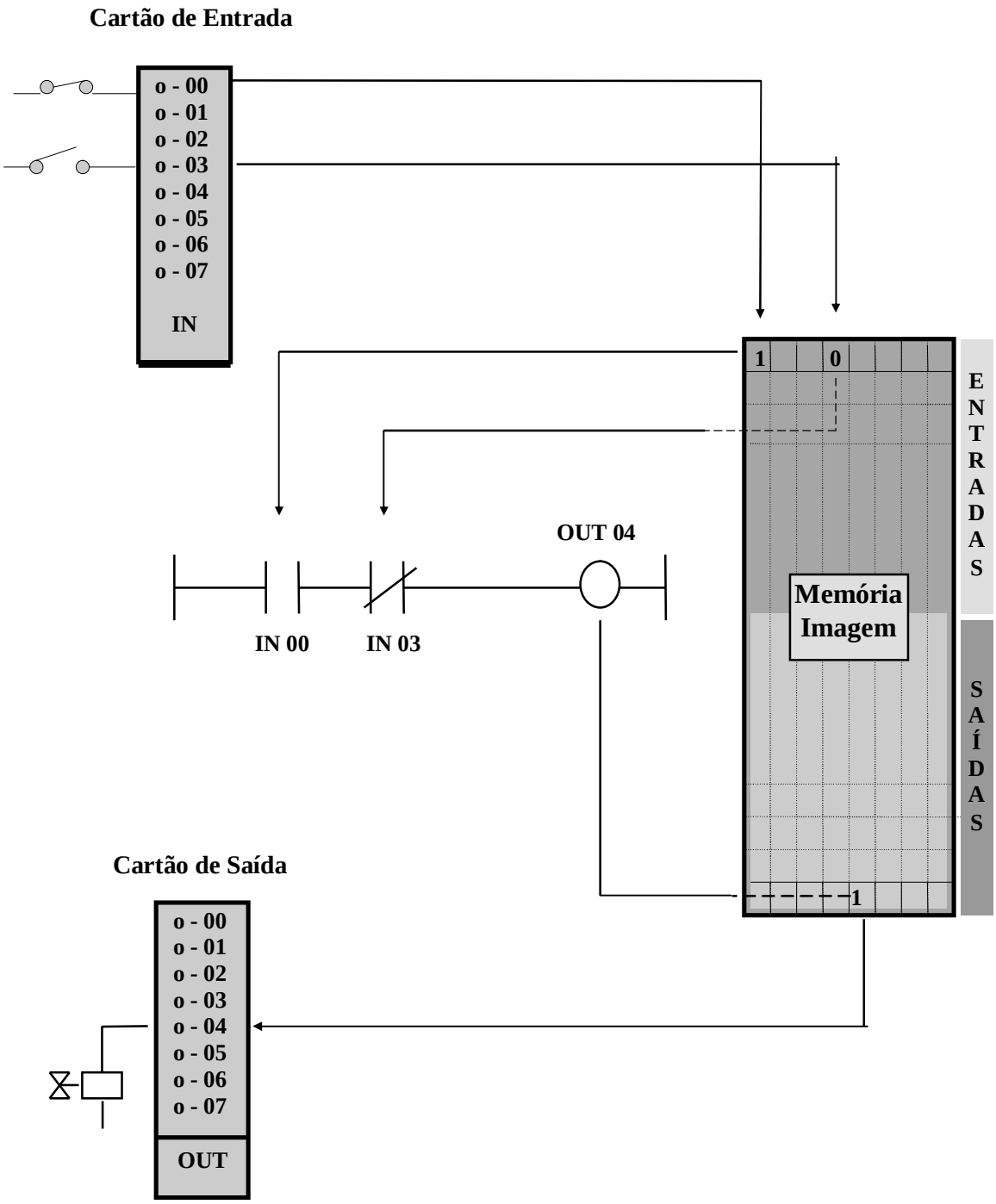
O termo *varredura* ou *scan*, são usados para um dar nome a um ciclo completo de operação (*loop*).

O tempo gasto para a execução do ciclo completo é chamado *Tempo de Varredura*, e depende do tamanho do programa do usuário, e a quantidade de pontos de entrada e saída.



Fluxograma de funcionamento de um CLP

Ciclo de Operação de um CLP



V. Linguagem de Programação

Na execução de tarefas ou resolução de problemas com dispositivos microprocessados, é necessária a utilização de uma linguagem de programação, através da qual o usuário se comunica com a máquina.

A linguagem de programação é uma ferramenta necessária para gerar o programa, que vai coordenar e sequenciar as operações que o microprocessador deve executar.

V.1. CLASSIFICAÇÃO

⇒ **Linguagem de baixo nível**
⇒ **Linguagem de alto nível**

LINGUAGEM DE BAIXO NÍVEL

Linguagem de Máquina

É a linguagem corrente de um microprocessador ou microcontrolador, onde as instruções são escritas em *código binário* (*bits 0 e 1*). Para minimizar as dificuldades de programação usando este código, pode-se utilizar também o *código hexadecimal*.

Código Binário

Endereço	*Conteúdo*
0000000000000000	00111110
0000000000000001	10000000
0000000000000010	11010011
0000000000000011	00011111
0000000000000100	00100001
0000000000000101	00000000
0000000000000111	01111110
0000000000001000	00100011
0000000000001001	10000110
0000000000001010	00111111
0000000000001011	00000001
0000000000001111	11011010
0000000000010000	00000000
0000000000010001	11011010

Código Hexadecimal

Endereço	*Conteúdo*
0000	**3E**
0001	**80**
0002	**D3**
0003	**1F**
0004	**21**
0005	**00**
0006	**10**
0007	**7E**
0008	**23**
0009	**86**
000A	**27**
000B	**D3**
000C	**17**
000D	**3F**

Cada item do programa, chama-se linha ou passo, representa uma instrução ou dado a ser operacionalizado.

Linguagem Assembler

Na linguagem assembler o programa é escrito com instruções abreviadas chamadas mnemônicos.

Endereço	*Conteúdo*
0000	**MVI A,80H**
0002	**OUT 1FH**
0004	**LXI ,1000H**
0007	**MOV A,M**
0008	**INX H**
0009	**ADD M**
000A	**DAA**
000B	**OUT 17H**
000D	**MVI A,1H**
000F	**JC 0031H**
0012	**XRA A**
0013	**OUT 0FH**
0015	**HLT**

Cada microprocessador ou microcontrolador possuem estruturas internas diferentes, portanto seus conjuntos de registros e instruções também são diferentes.

LINGUAGEM DE ALTO NÍVEL

É uma linguagem próxima da linguagem corrente utilizada na comunicação de pessoas.

Compiladores e Interpretadores

Quando um microcomputador utiliza uma linguagem de alto nível, é necessário a utilização de compiladores e interpretadores para traduzirem este programa para a linguagem de máquina.

Vantagem
Elaboração de programa em tempo menor, não necessitando conhecimento da arquitetura do microprocessador.

Desvantagem
Tempo de processamento maior do que em sistemas desenvolvidos em linguagens de baixo nível.

Exemplos de linguagens de alto nível

- Pascal
- C
- Fortran
- Cobol
- etc

VI. Programação de Controladores Programáveis

Normalmente podemos programar um controlador através de um software que possibilita a sua *apresentação* ao usuário em quatro formas diferentes:

- Diagrama de contatos;
- Diagrama de blocos lógicos (lógica booleana);
- Lista de instruções;
- Linguagem corrente.

Alguns CLPs, possibilitam a apresentação do programa do usuário em uma ou mais formas.

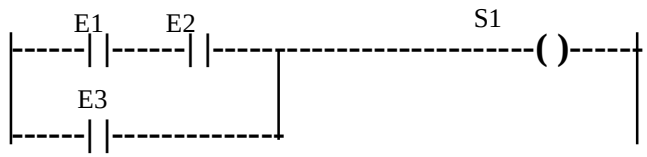
VI.1. DIAGRAMA DE CONTATOS

Também conhecida como:

- Diagrama de relés;
- Diagrama escada;
- Diagrama ñladderò.

Esta forma gráfica de apresentação está muito próxima a normalmente usada em diagrama elétricos.

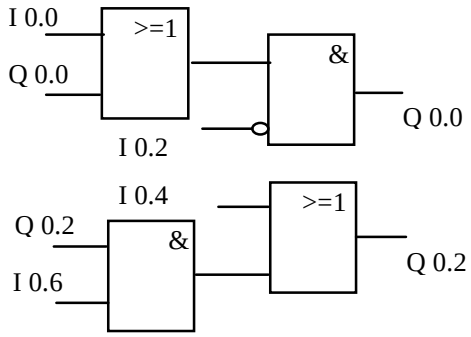
Exemplo:



VI.2. DIAGRAMA DE BLOCOS LÓGICOS

Mesma linguagem utilizada em lógica digital, onde sua representação gráfica é feita através das chamadas *portas lógicas*.

Exemplo:



VI.3. LISTA DE INSTRUÇÃO

Linguagem semelhante à utilizada na elaboração de programas para computadores.

Exemplo :

: A I 1.5
: A I 1.6
: O
: A I 1.4
: A I 1.3
: = Q 3.0

(I 1.5 . I 1.6) + (I 1.4 . I 1.3) = Q 3.0

VI.4. LINGUAGEM CORRENTE

É semelhante ao basic, que é uma linguagem popular de programação, e uma linguagem de programação de alto nível. Comandos típicos podem ser "fechar válvula A" ou "desligar bomba B", "ligar motor", "desligar solenóide",

VI.5. ANÁLISE DAS LINGUAGUES DE PROGRAMAÇÃO

Com o objetivo de ajudar na escolha de um sistema que melhor se adapte as necessidades de cada usuário, pode-se analisar as características das linguagens programação disponíveis de CLPs.

Esta análise se deterá nos seguintes pontos:

- Quanto a forma de programação;
- Quanto a forma de representação;
- Documentação;
- Conjunto de Instruções.

Quanto a Forma de Programação

.**Programação Linear** - programa escrito escrita em único bloco

.**Programação Estruturada** - Estrutura de programação que permite:

- Organização;
- Desenvolvimento de bibliotecas de rotinas utilitárias para utilização em vários programas;
- Facilidade de manutenção;
- Simplicidade de documentação e entendimento por outras pessoas além do autor do software.

Permite dividir o programa segundo critérios funcionais, operacionais ou geográficos.

Quanto a Forma de Representação

. Diagrama de Contatos;
. Diagrama de Blocos;
. Lista de Instruções.
Estes já citados anteriormente.

Documentação

A documentação é mais um recurso do editor de programa que de linguagem de programação. De qualquer forma, uma abordagem neste sentido torna-se cada vez mais importante, tendo em vista que um grande número de profissionais estão envolvidos no projeto de um sistema de automação que se utiliza de CLPs, desde sua concepção até a manutenção.

Quanto mais rica em comentários, melhor a documentação que normalmente se divide em vários níveis.

Conjunto de Instruções

É o conjunto de funções que definem o funcionamento e aplicações de um CLP.

Podem servir para mera substituição de comandos a relés:
- Funções Lógicas;
- Memorização;
- Temporização;
- Contagem.

Como também manipulação de variáveis analógicas:
- Movimentação de dados;
- Funções aritméticas.

Se funções complexas de algoritmos, comunicação de dados, interfaces homem-máquina, podem ser necessárias:
- Saltos controlados;
- Indexação de instruções;
- Conversão de dados;
- PID;
- sequenciadores;
- aritmética com ponto flutuante;
- etc.

VI.6. NORMALIZAÇÃO

Existe a tendência de utilização de um padrão de linguagem de programação onde será possível a intercambiabilidade de programas entre modelos de CLPs e até de fabricantes diferentes.

Esta padronização está de acordo com a norma IEC 1131-3, na verdade este tipo de padronização é possível utilizando-se o conceito de linguagem de alto nível, onde através de um chamado compilador, pode-se adaptar um programa para a linguagem de máquina de qualquer tipo de microprocessador, isto é, um programa padrão, pode servir tanto para o CLP de um fabricante A como de um fabricante B.

A norma IEC 1131-3 prevê três linguagens de programação e duas formas de apresentação. As linguagens são:

- ***Ladder Diagram*** - programação como esquemas de relés.
- ***Boolean Blocks*** - blocos lógicos representando portas ñEò, ñOUò, ñNegaçãoò, ñOu exclusivoò, etc.
- ***Structured Control Language*** (SCL) - linguagem que vem substituir todas as linguagens declarativas tais como linguagem de instruções, BASIC estruturado e inglês estruturado. Esta linguagem é novidade no mercado internacional e é baseada no Pascal.

As formas de representação são :

- Programação convencional;
- *Sequencial Function Chart* (SFC) - evolução do *graphcet* francês.

A grande vantagem de se ter o software normalizado é que em se conhecendo um conhece-se todos, economizando em treinamento e garantindo que, por mais que um fornecedor deixe o mercado, nunca se ficará sem condições de crescer ou repor equipamentos.

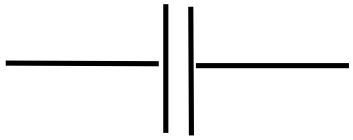
VII. Programação em Ladder

O diagrama ladder utiliza lógica de relé, com contatos (ou chaves) e bobinas, e por isso é a linguagem de programação de CLP mais simples de ser assimilada por quem já tenha conhecimento de circuitos de comando elétrico.

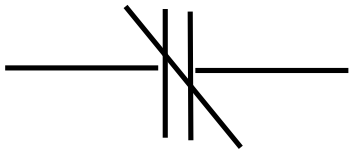
Compõe-se de vários circuitos dispostos horizontalmente, com a bobina na extremidade direita, alimentados por duas barras verticais laterais. Por esse formato é que recebe o nome de **ladder** que significa escada, em inglês.

Cada uma das linhas horizontais é uma sentença lógica onde os contatos são as entradas das sentenças, as bobinas são as saídas e a associação dos contatos é a lógica.

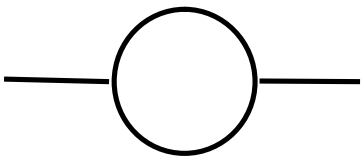
São os seguintes os símbolos:



CONTATO NORMALMENTE ABERTO



CONTATO NORMALMENTE FECHADO



BOBINA

No ladder cada operando (nome genérico dos contatos e bobinas no ladder) é identificado com um endereço da memória à qual se associa no CLP. Esse endereço aparece no ladder com um nome simbólico, para facilitar a programação, arbitrariamente escolhido pelo fabricante como os exemplos vistos a seguir.

Tabela de alguns CLPs X endereçamento

FABRICANTE	MODELO	E.D.	S.D.	E.A.	S.A.	BIT AUX.	PALAVRA	PALAVRA DO SISTEMA	CONTADOR / TEMPORIZADOR
GEFANUC	90-70 90-30 90-20 90-MICRO	%I1 a %I...	%Q1 a %Q...	%AI a %AI...	%AQ1 a %AQ... .	%M1 a %M... %T1 a %T...	%R1 a %R...	%S	%Rx x x+1 x+2 PARA CADA
ALLEN BRADLEY	SLC-500	I:SLOT.PONTO I:1/0 a I:...	O:SLOT.PONTO O:1/0 a O:...	I:SLOT.PONTO I:3.0 a I:3....	O:SLOT.PONTO O:3.0 a O:3....	B3:0/0 a B3:...	N7:0 a N7:...	S: R6:0 a R6:...	T4:0 A T4:... C5:0 A C5:...
ALTUS	AL500	R0 a R...	R60 a R...	-	-	A0 a A...	M0 a M...	-	M0 PARA CADA
ALTUS	PICOLLO	%E0.0 a %E...	%S2.0 a %S...	%M	%M	%A0.0 a %A...	%M0 a %M...		%M0 PARA CADA
FESTO	FPC101 FPC103	I0.0 a I...	O0.0 a O...	II0 a II3 OU IU0 a IU3	OU0 e OU1	F0.0 a F15.15	R0 a R64	FW0 a FW15	T0 a T31 C0 a C15

Outros tipos de endereçamento; 125/04 (1 = entrada, 2 = gaveta, 5 = número do cartão ou módulo, 04 = número do ponto), 013/01 (0 = saída, 1 = número da gaveta, 3 = número do módulo, 01 = número do ponto).

Nesta apostila os endereços serão identificados como:

E - para entrada digital;

EA - para entrada analógica;

S - para saída digital;

SA - para saída analógica.

A - para bobina auxiliar

O estado de cada operando é representado em um bit correspondente na memória imagem: este bit assume nível **1** se o operando estiver acionado e **0** quando desacionado.

*** As bobinas acionam o seu endereço**

Enquanto uma bobina com endereço de saída estiver acionada, um par de terminais no módulo de saída será mantido em condição de condução elétrica.

*** Os contatos se acionam pelo endereço que os identifica.**

.

Os contatos endereçados como entrada se acionam enquanto seu respectivo par de terminais no módulo de entrada é acionado: fecham-se se forem NA e abrem-se se forem NF.

Com relação ao que foi exposto acima sobre os contatos endereçados como entrada, **os que tiverem por finalidade acionar ou energizar uma bobina deverão ser do mesmo tipo do contato externo que aciona seu respectivo ponto no módulo de entrada.**

Já os que forem usados para desacionar ou desenergizar uma bobina devem ser de tipo contrário do contato externo que os aciona. Abaixo vê-se um quadro elucidativo a esse respeito.

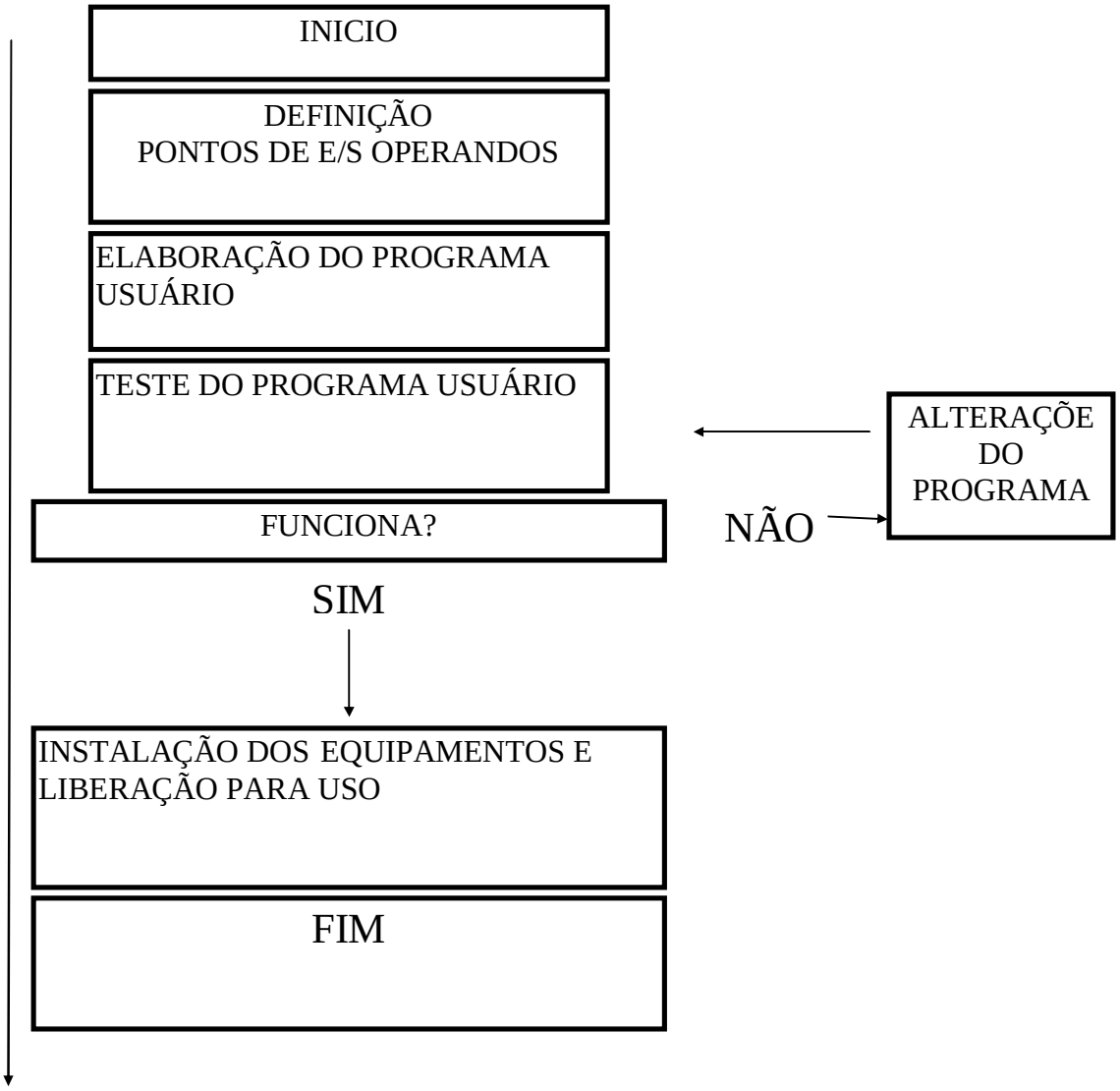
	Se a chave externa for	o contato no ladder deve ser
Para ligar	**NA**	**NA**
	NF	**NF**
Para desligar	**NA**	**NF**
	NF	**NA**

Percebe-se pois que pode ser usada chave externa de qualquer tipo, desde que no ladder se utilize o contato de tipo conveniente. Mesmo assim, por questão de segurança, não se deve utilizar chave externa NF para ligar nem NA para desligar.

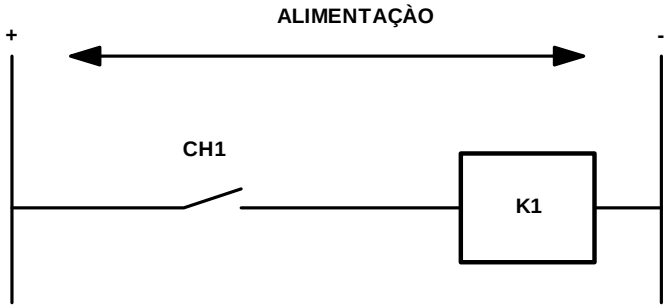
VII.1. DESENVOLVIMENTO DO PROGRAMA LADDER

Após a definição da operação de um processo onde são geradas as necessidades de seqüenciamento e/ou intertravamento, esses dados e informações são passados sob forma de diagrama lógico, diagrama funcional ou matriz de causas e efeitos e a partir daí o programa é estruturado.

Abaixo vêem-se os passos para a automação de um processo ou equipamento.

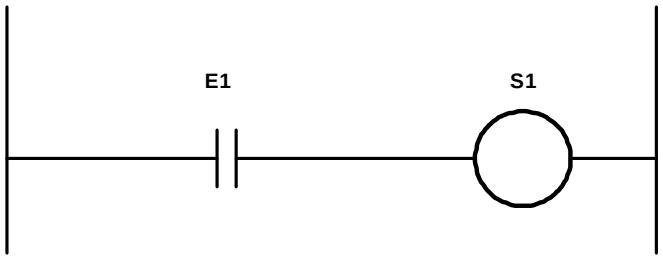


A lógica de diagrama de contatos do CLP assemelha-se à de relés. Para que um relê seja energizado, necessita de uma continuidade elétrica, estabelecida por uma corrente elétrica.



Ao ser fechada a CH1, a bobina K1 será energizada, pois será estabelecida uma continuidade entre a fonte e os terminais da bobina.

O programa equivalente do circuito anterior, na linguagem ladder, será o seguinte.

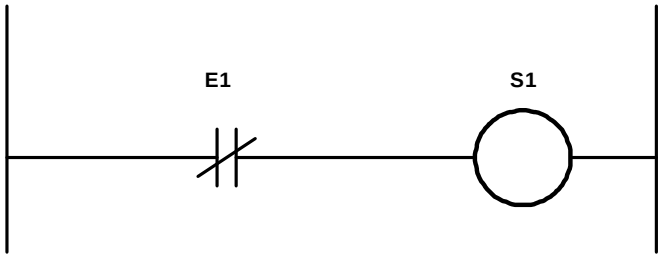


Analisando os módulos de entrada e saída do CLP, quando o dispositivo ligado à entrada digital E1 fechar, este acionará o contato E1, que estabelecerá uma continuidade de forma a acionar a bobina S1, consequentemente o dispositivo ligado à saída digital S1 será acionado.

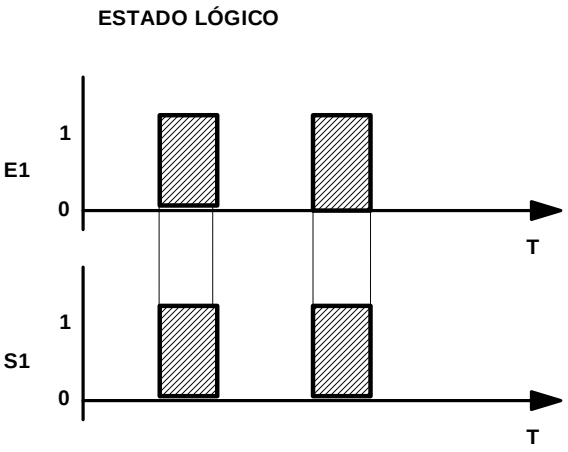
Uma prática indispensável é a elaboração das tabelas de alocação dos dispositivos de entrada/saída. Esta tabela é constituída do nome do elemento de entrada/saída, sua localização e seu endereço de entrada/saída no CLP. Exemplo:

DISPOSITIVO	LOCALIZAÇÃO	ENDEREÇO
PSL - 100	Topo do tanque pressurizado 2	E1
TT - 400	Saída do misturador	EA1
FS	Saída de óleo do aquecedor	E2
SV	Ao lado da válvula FV400	S1

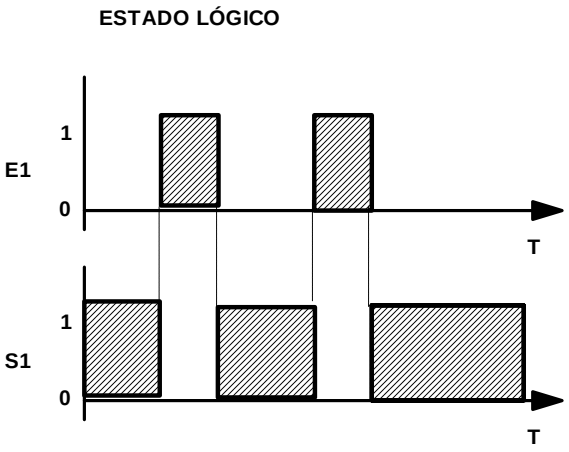
O NF é um contado de negação ou inversor, como pode ser visto no exemplo abaixo que é similar ao programa anterior substituindo o contato NA por um NF.



Analisando os módulos de entrada e saída, quando o dispositivo ligado a entrada digital E1 abrir, este desacionará o contato E1, este por ser NF estabelecerá uma continuidade de forma a acionar a bobina S1, consequentemente o dispositivo ligado à saída digital S1 será acionado. A seguir temos o gráfico lógico referente aos dois programas apresentados anteriormente.



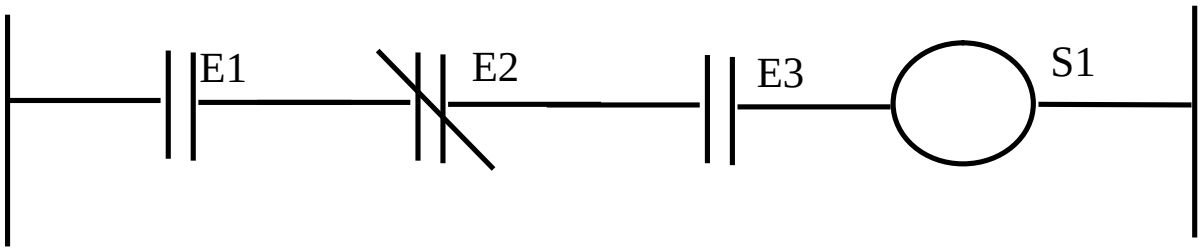
CIRCUITO UTILIZANDO E1 NORMALMENTE ABERTO



CIRCUITO UTILIZANDO E1 NORMALMENTE FECHADO

VII.1.1 ASSOCIAÇÃO DE CONTATOS NO LADDER.

No ladder se associam contatos para criar as lógicas **E** e **OU** com a saída.
Os contatos em série executam a lógica **E,** pois a bobina só será acionada *quando todos os contatos estiverem fechados*

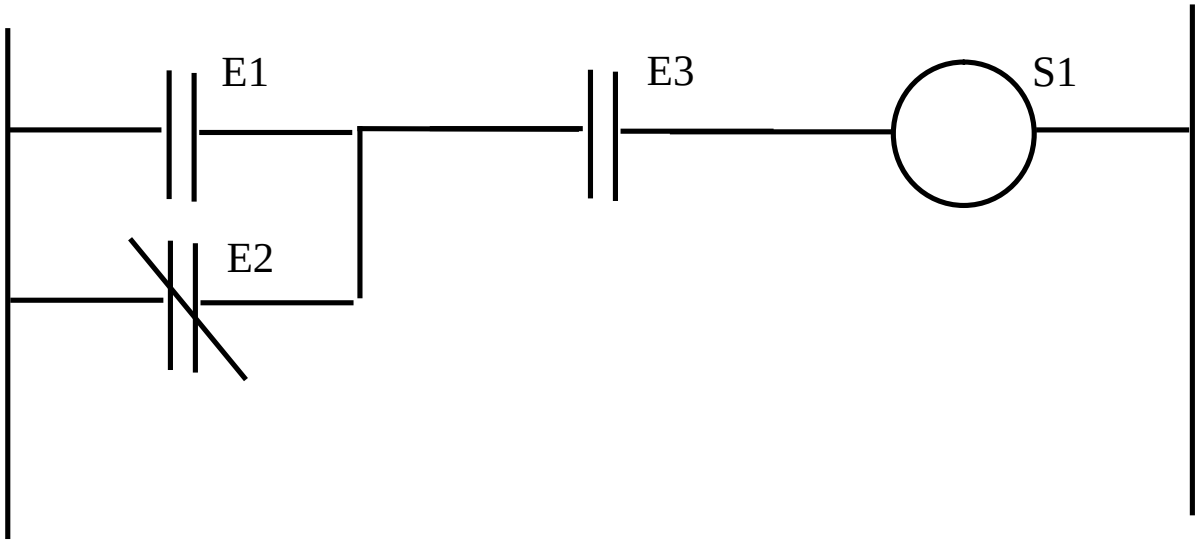


A saída S1 será acionada quando:
E1 estiver acionada **E**
E2 estiver não acionada **E**
E3 estiver acionada
Em álgebra booleana $S=E1*\overline{E2}*E3$

A lógica **OU** é conseguida com a associação paralela, acionando a saída desde que pelo menos um dos ramos paralelos estejam fechados

A saída S1 será acionada se
E1 for acionada OU
E2 não for acionada OU
E3 for acionada. O que equivale a lógica booleana.
$S1=E1+\overline{E2}+E3$

Com associações mistas criam-se condições mais complexas como a do exemplo a seguir



Neste caso a saída é acionada quando

E3 for acionada & E1 for acionada

OU

E3 for acionada & E2 não for acionada

Em lógica booleana $S1=E3 * (E1 + \overline{E2})$

VII.1.2. INSTRUÇÕES

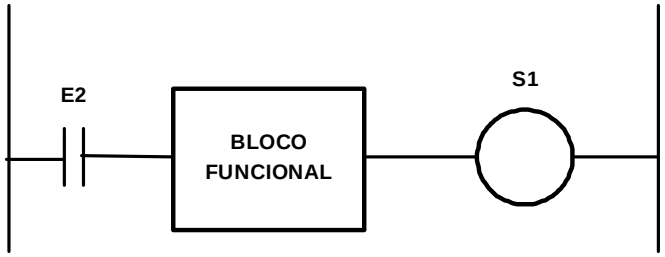
Na UCP o programa residente possui diversos tipos de blocos de funções. Na listagem a seguir apresentamos alguns dos mais comuns:

- contador;
- temporização de energização;
- temporização de desenergização;
- adição de registros;
- multiplicação de registros;
- divisão de registros;
- extração de raiz quadrada;
- bloco OU lógico de duas tabelas;
- bloco E lógico de duas tabelas;
- ou exclusivo lógico de duas tabelas;
- deslocar bits através de uma tabela-direita;
- deslocar bits através de uma tabela-esquerda;
- mover tabela para nova localização;
- mover dados para memória EEPROM;
- mover inverso da tabela para nova localização;
- mover complemento para uma nova localização;
- mover valor absoluto para uma nova localização;
- comparar valor de dois registros;
- ir para outra seqüência na memória;
- executar sub-rotina na memória;
- converter A/D e localizar em um endereço;
- converter D/A um dado localizado em um endereço;
- executar algoritmo PID;
- etc.

VII.1.3. INSTRUÇÕES BÁSICAS

As instruções básicas são representadas por blocos funcionais introduzidos na linha de programação em lógica ladder. Estes blocos funcionais podem se apresentar de formas diferentes de um CLP para outro, mas a filosofia de funcionamento é invariável. Estes blocos auxiliam ou complementam o controle do equipamento, introduzindo na lógica ladder instruções como de temporização, contagem, soma, divisão, subtração, multiplicação, PID, conversão BCD/Decimal, conversão Decimal/BCD, raiz quadrada, etc.

FUNCIONAMENTO DOS PRINCIPAIS BLOCOS

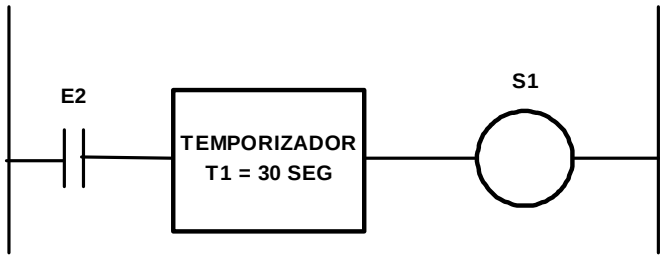


O bloco funcional possui pontos de entrada (localizados à esquerda) e pontos de saída (localizados à direita do bloco), também possui campos de entrada de informações como; número do registro, memória, ponto de entrada analógico, bit de saída, bit de entrada, ponto de saída analógico, constantes, etc.

As instruções seguintes será explicadas supondo o byte de oito bits. A análise para o byte de dezesseis bits é exatamente a mesma.

INSTRUÇÃO DE TEMPORIZAÇÃO

O temporizador conta o intervalo de tempo transcorrido a partir da sua habilitação até este se igualar ao tempo preestabelecido. Quando a temporização estiver completa esta instrução eleva a nível 1 um bit próprio na memória de dados e aciona o operando a ela associado.



Segundo exemplo, quando E1 for acionada, o temporizador será habilitado e imediatamente após 30 segundos a saída S1 será acionada. Quando E1 for desacionada, o temporizador será desabilitado, ou desenergizado, desacionando a saída S1. Em alguns casos, esta instrução apresenta duas entradas uma de habilitação da contagem e outra para zeramento ou reset da saída.

Para cada temporizador destina-se um endereço de memória de dados onde o valor prefixado será armazenado.

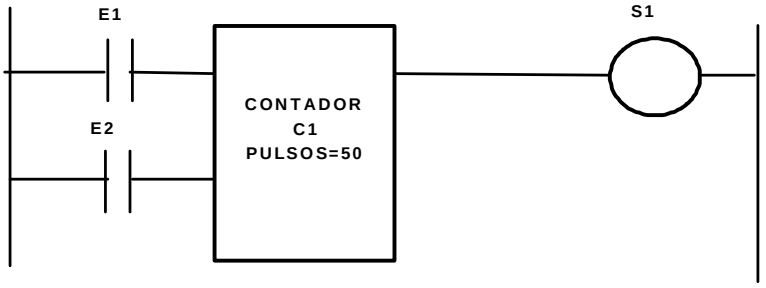
Na memória de dados do CLP, o temporizador ocupa três bytes para o controle. O primeiro byte reservado para o dado prefixado, o segundo byte reservado para a temporização e o terceiro byte reservado para os bits de controle da instrução temporizador.

Os temporizadores podem ser TON (temporiza no acionamento) e TOFF (temporiza no desacionamento).

1º byte = valor prefixado de 30 seg.
2º byte = tempo transcorrido
3º byte = bits de controle D.E. (bit de entrada) e D.S. (bit de saída).

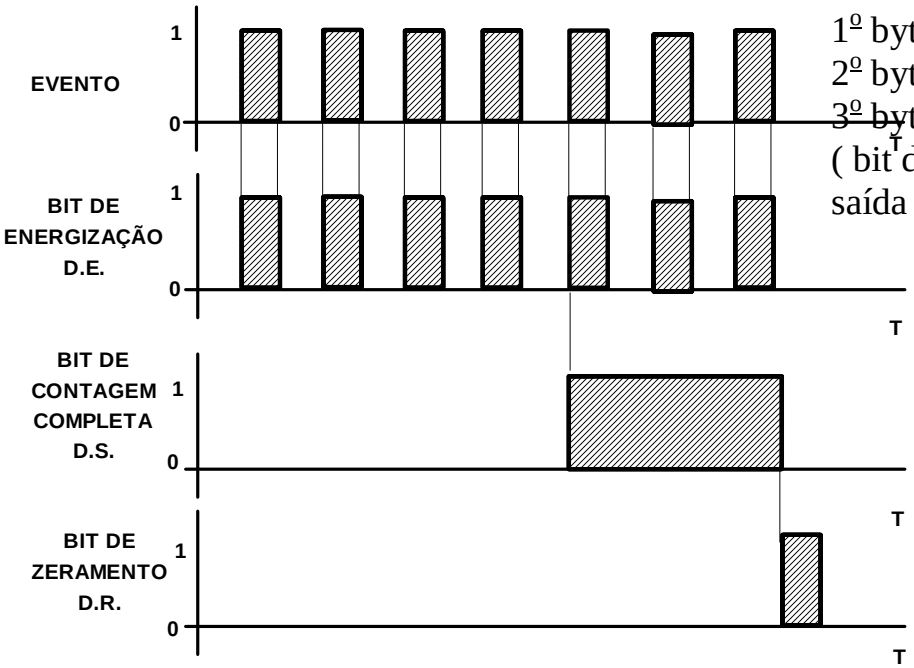
INSTRUÇÃO DE CONTAGEM

O contador conta o número de eventos que ocorre e deposita essa contagem em um byte reservado. Quando a contagem estiver completa, ou seja , igual ao valor prefixado, esta instrução energiza um bit de contagem completa. A instrução contador é utilizada para energizar ou desenergizar um dispositivo quando a contagem estiver completa.



Para cada contador destina-se um endereço de memória de dados onde o valor prefixado será armazenado.

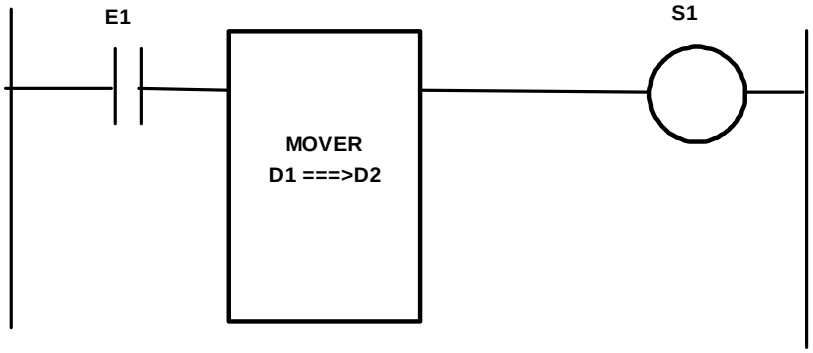
Na memória de dados do CLP, o contador ocupa três bytes para o controle. O primeiro byte reservado para o dado prefixado, o segundo byte reservado para a contagem e o terceiro byte reservado para os bits de controle da instrução contador.



1º byte = valor prefixado de 50
2º byte = contagem
3º byte = bits de controle D.E. (bit de entrada), D.S. (bit de saída) e D.R. (bit de reset).

INSTRUÇÃO MOVER

A instrução mover transfere dados de um endereço de memória para outro endereço de memória, manipula dados de endereço para endereço, permitindo que o programa execute diferentes funções com o mesmo dado.



Abaixo temos cinco endereços da memória de dados do CLP. Observe que o dado de D1 é distinto de D2.

	B7	B6	B5	B4	B3	B2	B1	B0
D1	0	0	0	0	1	1	1	1
D2	0	0	1	1	0	0	0	0
D3	0	0	0	0	1	0	0	0
D4	1	1	1	0	0	1	0	0
D5	1	0	0	0	0	1	1	1

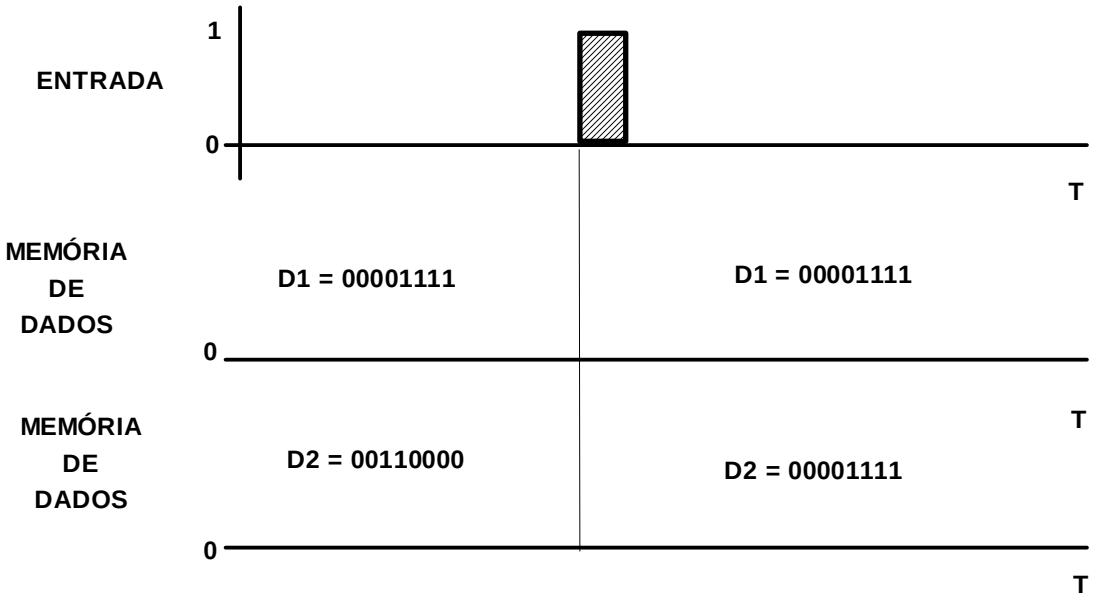
Supondo que a instrução mover tenha sido acionada e que a movimentação será de D1 para D2.

	B7	B6	B5	B4	B3	B2	B1	B0
D1	0	0	0	0	1	1	1	1
D2	0	0	0	0	1	1	1	1
D3	0	0	0	0	1	0	0	0
D4	1	1	1	0	0	1	0	0
D5	1	0	0	0	0	1	1	1

Observe que o conteúdo de D2 foi alterado. No momento em que a instrução mover for desacionada, o dado de D2 permanecerá o mesmo.

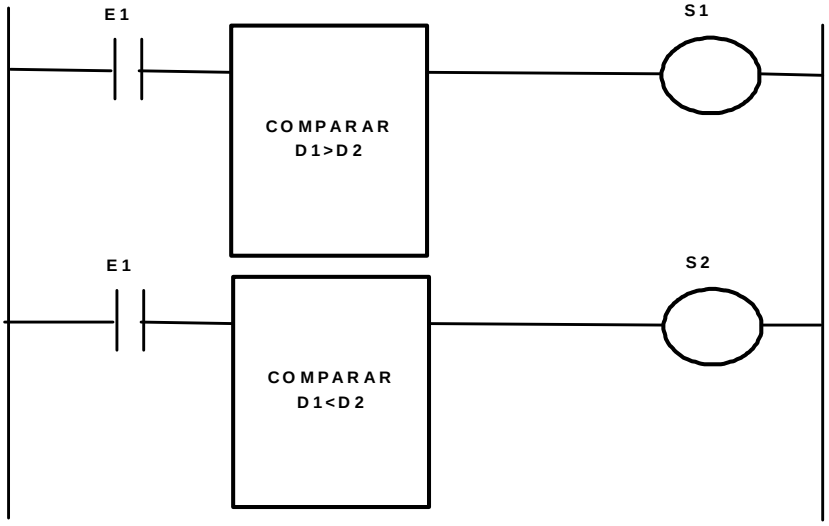
Enquanto E1 estiver acionada o dado será movido uma vez a cada ciclo de varredura, portanto E1 deve ser acionado e desacionado rapidamente.

Temos o gráfico que ilustra antes e depois do acionamento de E1 para a instrução mover.

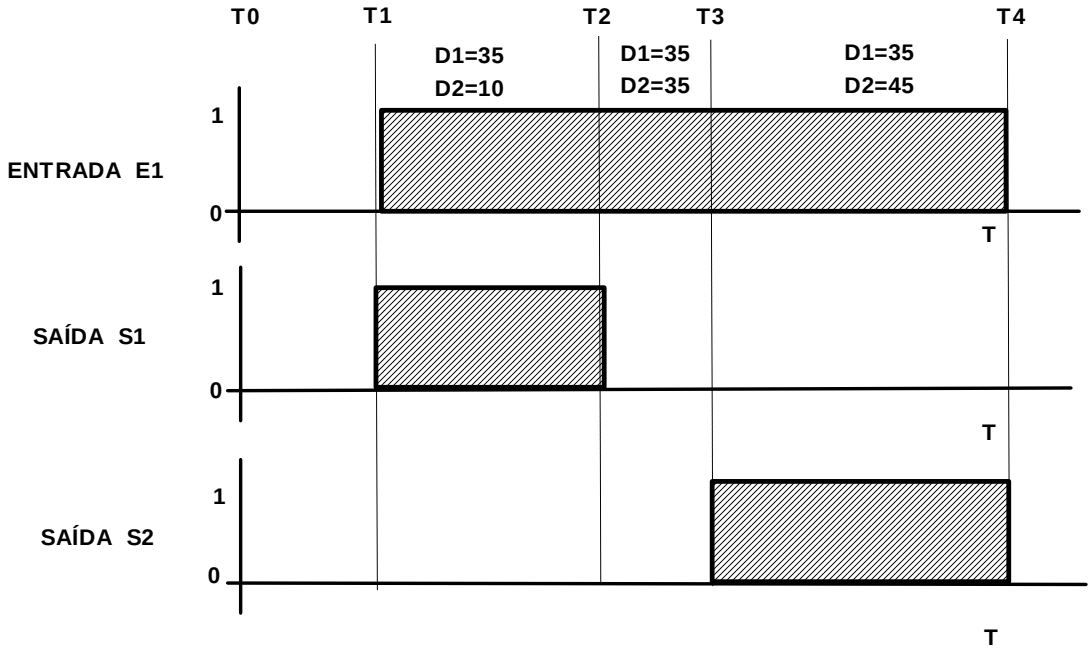


INSTRUÇÃO COMPARAR

A instrução comparar verifica se o dado de um endereço é igual, maior, menor, maior/igual ou menor/igual que o dado de um outro endereço, permitindo que o programa execute diferentes funções baseadas em um dado de referência.



No exemplo, quando a entrada E1 for acionada as duas instruções de comparação serão acionadas, se D1 for maior que D2 o bit de saída S1 será acionado, se D1 for menor que D2 o bit de saída S2 será acionado. A comparação só existirá se a entrada E1 estiver acionada, caso contrário as duas saídas S1 e S2 serão desacionadas.



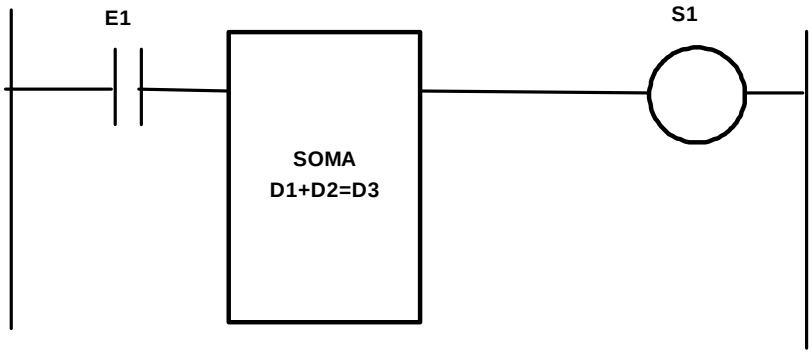
Observe o gráfico acima, entre T0 e T1 a entrada E1 está desativada, logo não há comparação e as saídas S1 e S2 são nulas. Entre T1 e T2 o dado D1 se encontra com valor maior que D2, logo a instrução de comparação ativa a saída S1. Entre T2 a T3 o dado D1 é igual a D2, como não há instrução de igualdade as saídas estarão desativadas. Entre T3 a T4 o dado D1 é menor que D2, logo a saída S2 será ativada, a partir de T4 a entrada E1 foi desacionada, portanto as comparações são desativadas e as saídas irão para estado lógico ñ0ò.

A mesma análise é válida para a instrução igual a, maior igual a e menor igual a.

VII.1.4. INSTRUÇÕES MATEMÁTICAS

INSTRUÇÃO SOMA

Permite somar valores na memória quando habilitado. Nesta instrução podem-se usar os conteúdos de um contador, temporizador, byte da memória imagem, byte da memória de dados.



Nesta instrução de programa, quando E1 for acionada, a soma do dado 1 com o dado 2 será depositado no dado 3, portanto o conteúdo do dado 3 não deverá ter importância. Caso o conteúdo do dado 3 seja importante, o mesmo deve ser movido para um outro endereço ou o resultado da soma depositado em outro endereço.

Enquanto E1 estiver acionado o dado D1 será somado com D2 e depositado no dado D3 a cada ciclo de varredura, portanto E1 deve ser acionado e desacionado rapidamente.

Abaixo temos cinco endereços da memória de dados do CLP.

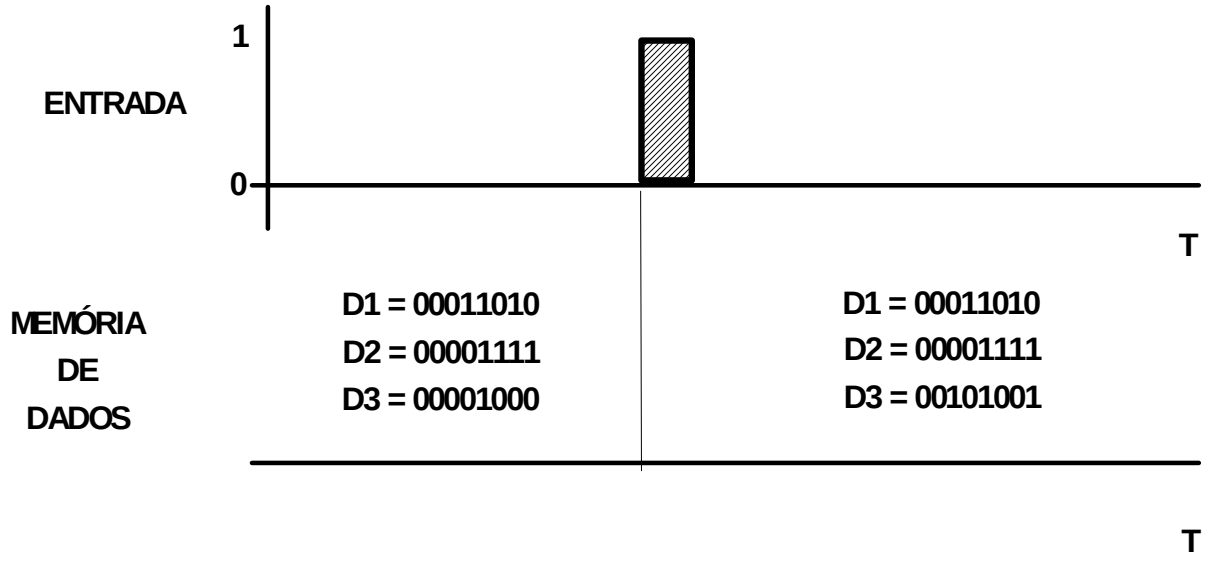
	B7	B6	B5	B4	B3	B2	B1	B0
D1	0	0	0	1	1	0	1	0
D2	0	0	0	0	1	1	1	1
D3	0	0	0	0	1	0	0	0
D4	1	1	1	0	0	1	0	0
D5	1	0	0	0	0	1	1	1

Supondo que a instrução somar tenha sido acionada e que a soma será de D1 e D2 em D3.

D1 equivale em decimal a 26 e D2 a 15, a soma resultará 41 no D3.

	B7	B6	B5	B4	B3	B2	B1	B0
D1	0	0	0	1	1	0	1	0
D2	0	0	0	0	1	1	1	1
D3	0	0	1	0	1	0	0	1
D4	1	1	1	0	0	1	0	0
D5	1	0	0	0	0	1	1	1

Observe que o conteúdo de D3 foi alterado, no momento em que a instrução soma for desacionada, os dados de D1 e D2 permanecerão os mesmos.

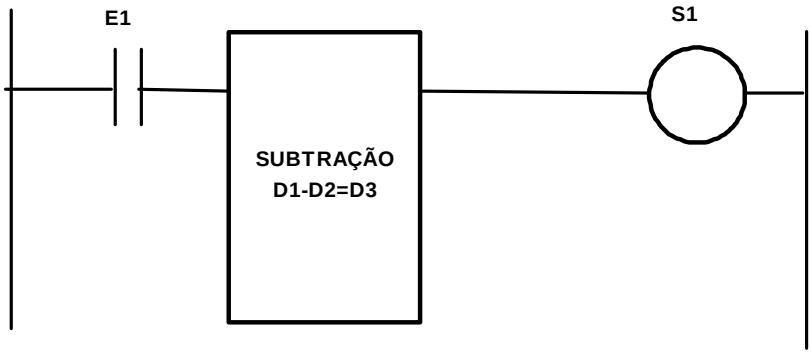


A saída S1 será acionada quando a soma for concluída.

Caso o resultado da soma não ultrapasse o limite máximo (overflow), a saída S1 será acionada. Em alguns casos o um bit, do byte de controle da instrução soma, assume valor lógico ñ1ò, determinando o estouro da capacidade. Através deste bit e possível de se determinar quando a soma ultrapassou ou não o valor máximo.

INSTRUÇÃO SUBTRAÇÃO

Permite subtrair valores na memória quando habilitado. Nesta instrução podem-se usar os conteúdo de um contador, temporizador, byte da memória imagem, byte da memória de dados.



Nesta instrução de programa, quando E1 for acionada, a subtração do dado 1 com o dado 2 será depositado no dado 3, portanto o conteúdo do dado 3 não deverá ter importância. Caso o conteúdo do dado 3 seja importante, o mesmo deve ser movido para um outro endereço ou o resultado da soma depositado em outro endereço.

Enquanto E1 estiver acionado o dado D1 será subtraído do dado D2 e depositado no dado D3 a cada ciclo de varredura, portanto E1 deve ser acionado e desacionado rapidamente.

Abaixo vêm-se cinco endereços da memória de dados do CLP.

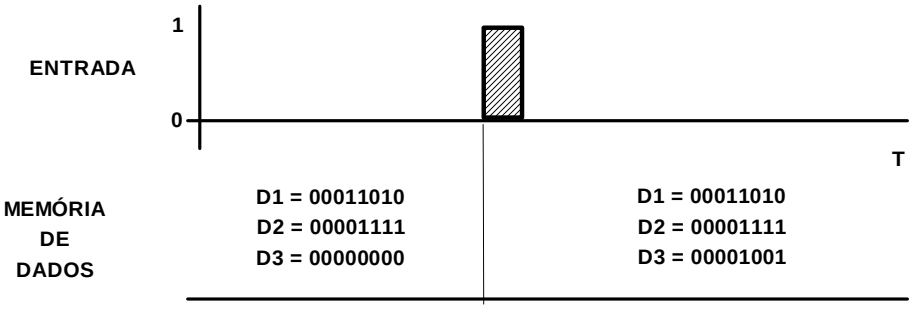
	B7	B6	B5	B4	B3	B2	B1	B0
D1	0	0	0	1	1	0	1	0
D2	0	0	0	0	1	1	1	1
D3	0	0	0	0	0	0	0	0
D4	1	1	1	0	0	1	0	0
D5	1	0	0	0	0	1	1	1

Supondo que a instrução subtração tenha sido acionada e que a subtração será de D1 menos D2 em D3.

D1 equivale em decimal a 26 e D2 a 15, a subtração resultará 9 no D3.

	B7	B6	B5	B4	B3	B2	B1	B0
D1	0	0	0	1	1	0	1	0
D2	0	0	0	0	1	1	1	1
D3	0	0	0	0	1	0	0	1
D4	1	1	1	0	0	1	0	0
D5	1	0	0	0	0	1	1	1

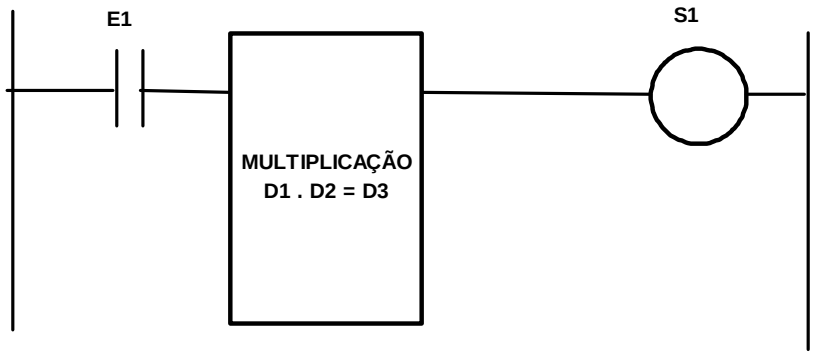
Observe que o conteúdo de D3 foi alterado, no momento em que a instrução soma for desacionada, os dados de D1 e D2 permanecerão os mesmos.



Caso o resultado da subtração possua sinal negativo (underflow), a saída S1 será acionada. Em alguns casos o um bit, do byte de controle da instrução subtração, assume valor lógico ñ1ò. Através deste bit e possível de se determinar quando a subtração resultou positivo ou negativo.

INSTRUÇÃO MULTIPLICAÇÃO

Permite multiplicar valores na memória se a condição for verdadeira.



Observe os cinco endereços do mapa de memória apresentado.

	B7	B6	B5	B4	B3	B2	B1	B0
D1	0	0	0	1	1	0	1	0
D2	0	0	0	0	0	1	1	1
D3	0	0	0	0	0	0	0	0
D4	1	1	1	0	0	1	0	0
D5	1	0	0	0	0	1	1	1

Supondo que a instrução multiplicação tenha sido acionada por E1 e que a multiplicação será de D1 por D2 em D3.

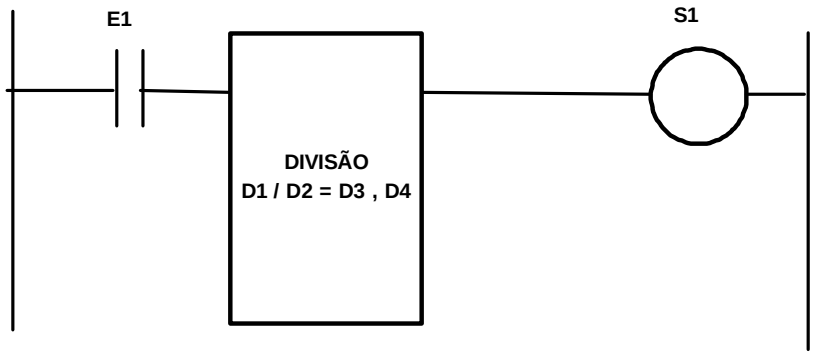
D1 equivale em decimal a 26 e D2 a 7, a multiplicação resultará 182 no D3.

	B7	B6	B5	B4	B3	B2	B1	B0
D1	0	0	0	1	1	0	1	0
D2	0	0	0	0	0	1	1	1
D3	1	0	1	1	0	1	1	0
D4	1	1	1	0	0	1	0	0
D5	1	0	0	0	0	1	1	1

Quando a entrada E1 for acionada, a multiplicação do dado D1 pelo dado D2 será depositada no conteúdo do dado D3.

INSTRUÇÃO DIVISÃO

Permite dividir valores na memória quando habilitado.



Observe os cinco endereços do mapa de memória apresentado.

	B7	B6	B5	B4	B3	B2	B1	B0
D1	0	0	1	1	0	0	1	0
D2	0	0	0	0	0	1	0	0
D3	0	0	0	0	0	0	0	0
D4	1	1	1	0	0	1	0	0
D5	1	0	0	0	0	1	1	1

Supondo que a instrução divisão tenha sido acionada por E1 e que a divisão será de D1 por D2 em D3, D4.

D1 equivale em decimal a 50 e D2 a 4, a divisão resultará 12,5 no D3, D4.

	B7	B6	B5	B4	B3	B3	B2	B1
D1	0	0	1	1	0	0	1	0
D2	0	0	0	0	0	1	0	0
D3	0	0	0	0	1	1	0	0
D4	0	0	0	0	0	1	0	1
D5	1	0	0	0	0	1	1	1

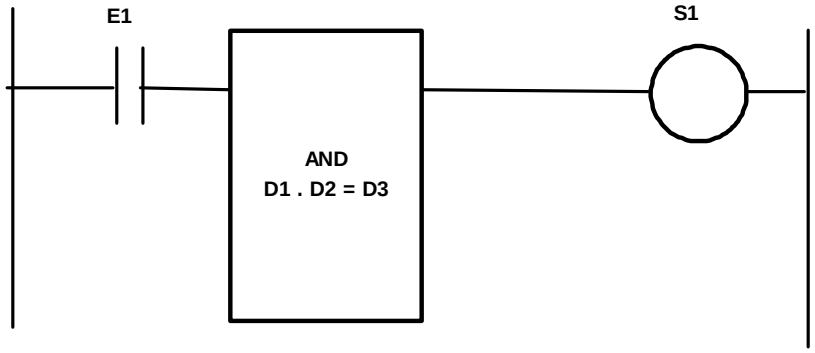
Quando a entrada E1 for acionada, a divisão do dado D1 pelo dado D2 será depositada no conteúdo do dado D3, D4.

VII.1.5. INSTRUÇÕES LÓGICAS

Estas instruções destinam-se à comparação lógica entre bytes. São recursos disponíveis para os programadores, podendo serem empregadas na análise de byte e diagnose de dados.

INSTRUÇÃO AND

Permite executar função AND com valores da memória quando habilitada .



Observe os cinco endereços do mapa de memória apresentado.

	B7	B6	B5	B4	B3	B2	B1	B0
D1	0	1	0	1	1	0	1	0
D2	0	1	0	0	0	1	1	1
D3	0	0	0	0	0	0	0	0
D4	1	1	1	0	0	1	0	0
D5	1	0	0	0	0	1	1	1

Supondo que a instrução AND tenha sido acionada por E1 e que a instrução será de D1 and D2 em D3.

Observe a tabela verdade abaixo e verifique o resultado da analise AND entre os dois bytes D1 e D2.

E1	E2	SAÍDA
0	0	0
0	1	0
1	0	0
1	1	1

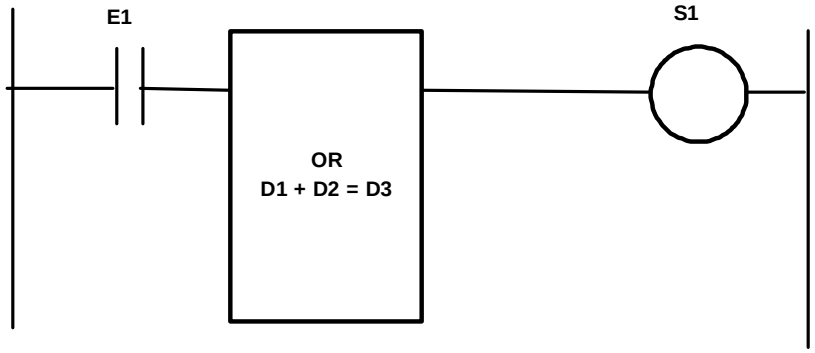
E1 e E2 são as entradas e SAÍDA é o resultado.

	B7	B6	B5	B4	B3	B2	B1	B0
D1	0	1	0	1	1	0	1	0
D2	0	1	0	0	0	1	1	1
D3	0	1	0	0	0	0	1	0
D4	1	1	1	0	0	1	0	0
D5	1	0	0	0	0	1	1	1

Quando a entrada E1 for acionada, a instrução do dado D1 and dado D2 será depositada no conteúdo do dado D3.

INSTRUÇÃO OR

Permite executar função OU com valores da memória quando habilitada analisar valores na memória quando habilitada.



Observe os cinco endereços do mapa de memória apresentado.

	B7	B6	B5	B4	B3	B2	B1	B0
D1	0	1	0	1	1	0	1	0
D2	0	1	0	0	0	1	1	1
D3	0	0	0	0	0	0	0	0
D4	1	1	1	0	0	1	0	0
D5	1	0	0	0	0	1	1	1

Supondo que a instrução OR tenha sido acionada por E1 e que a instrução será de D1 or D2 em D3.

Observe a tabela verdade abaixo e verifique o resultado da analise OR entre os dois bytes D1 e D2.

E1	E2	SAÍDA
0	0	0
0	1	1
1	0	1
1	1	1

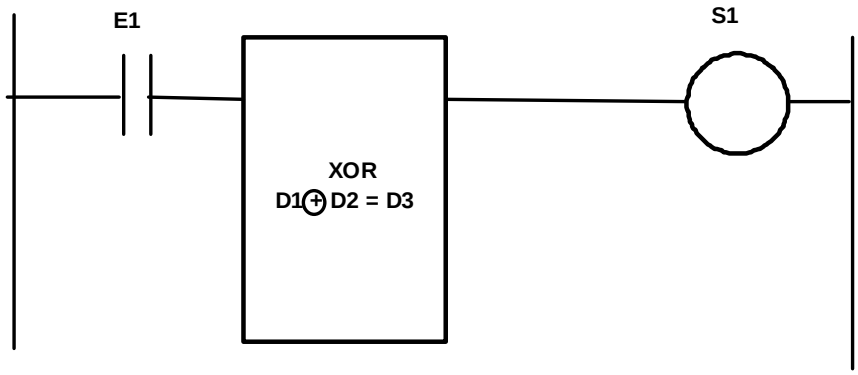
E1 e E2 são as entradas e SAÍDA é o resultado.

	B7	B6	B5	B4	B3	B2	B1	B0
D1	0	1	0	1	1	0	1	0
D2	0	1	0	0	0	1	1	1
D3	0	1	0	1	1	1	1	1
D4	1	1	1	0	0	1	0	0
D5	1	0	0	0	0	1	1	1

Quando a entrada E1 for acionada, a instrução do dado D1 or dado D2 será depositada no conteúdo do dado D3.

INSTRUÇÃO XOR

Permite executar função ou exclusivo com valores da memória quando habilitada.



Observe os cinco endereços do mapa de memória apresentado.

	B7	B6	B5	B4	B3	B2	B1	B0
D1	0	1	0	1	1	0	1	0
D2	0	1	0	0	0	1	1	1
D3	0	0	0	0	0	0	0	0
D4	1	1	1	0	0	1	0	0
D5	1	0	0	0	0	1	1	1

Supondo que a instrução XOR (ou exclusivo) tenha sido acionada por E1 e que a instrução será de D1 xor D2 em D3.

Observe a tabela verdade abaixo e verifique o resultado da análise xor entre os dois bytes D1 e D2.

E1	E2	SAÍDA
0	0	0
0	1	1
1	0	1
1	1	0

E1 e E2 são as entradas e SAÍDA é o resultado.

	B7	B6	B5	B4	B3	B2	B1	B0
D1	0	1	0	1	1	0	1	0
D2	0	1	0	0	0	1	1	1
D3	0	0	0	1	1	1	0	1
D4	1	1	1	0	0	1	0	0
D5	1	0	0	0	0	1	1	1

Quando a entrada E1 for acionada, a instrução do dado D1 xor dado D2 será depositada no conteúdo do dado D3.

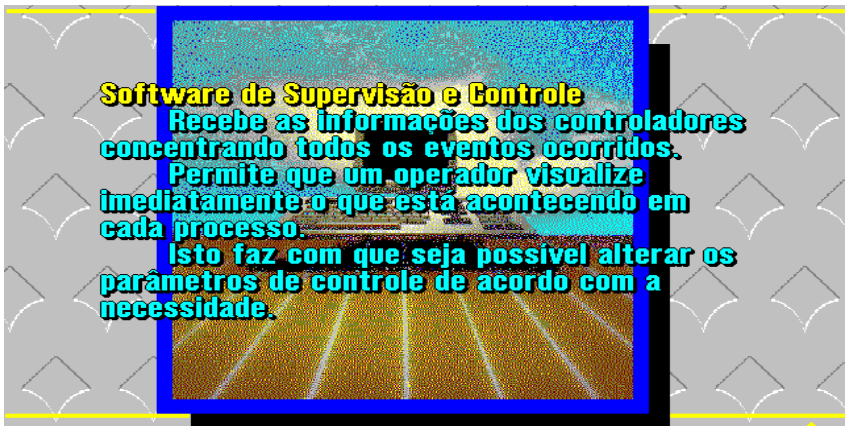
Obviamente estas são apenas algumas instruções que a programação ladder dispões. Uma série de outros recursos são disponíveis em função da capacidade do CLP em questão.

As instruções apresentadas servirão como base para o entendimento das instruções de programação ladder de qualquer CLP, para tal conte e não dispense o auxílio do manual ou help on-line quando disponível no software de programação.

A utilização do software de programação é uma questão de estudo e pesquisa, uma vez que o layout de tela e comandos não são padronizados.

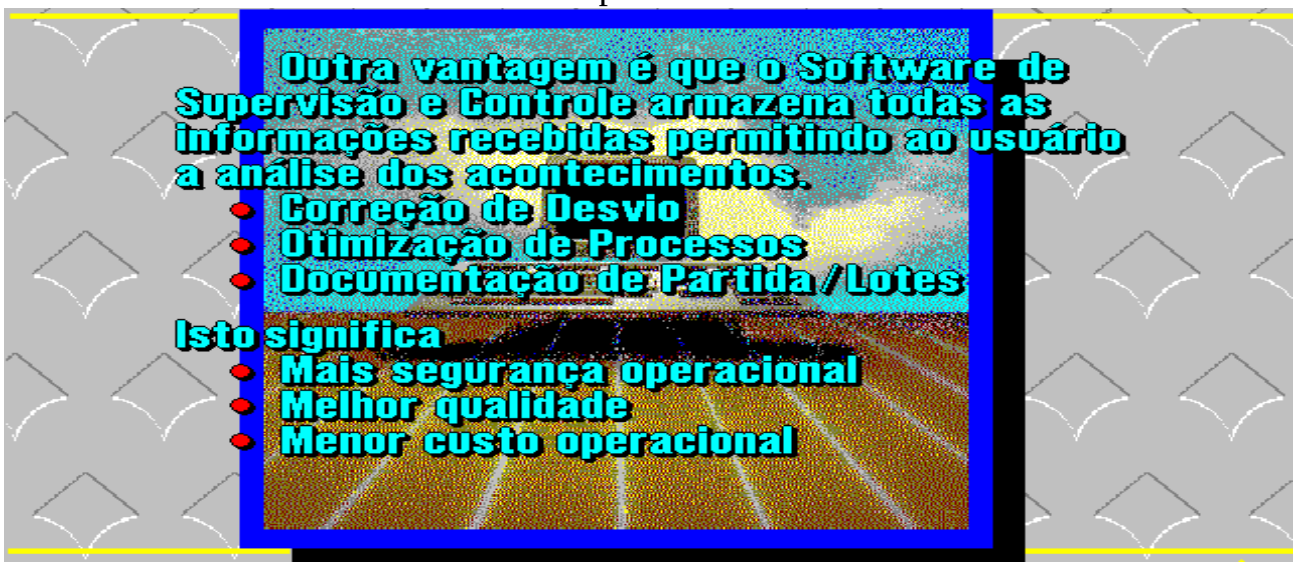
VIII. Noções de Sistema Supervisório ï Intouch.

A maior preocupação das empresas é aumentar a produtividade, com excelente qualidade, para tornar-se mais eficaz, flexível, competitiva e, sobretudo, mais lucrativa. Desse modo, investir em tecnologias de ponta e soluções sofisticadas é o primeiro passo para alcançar esse objetivo e, consequentemente, conquistar o mercado.


Software de Supervisão e Controle
Recebe as informações dos controladores concentrando todos os eventos ocorridos.
Permite que um operador visualize imediatamente o que está acontecendo em cada processo.
Isto faz com que seja possível alterar os parâmetros de controle de acordo com a necessidade.

Com o passar dos tempos, o advento do microprocessador tornou o mundo mais fácil de se viver. A utilização de microcomputadores e computadores no dia a dia nos possibilitou comodidade e rapidez.

Na indústria tem-se a necessidade de centralizar as informações de forma a termos o máximo possível de informações no menor tempo possível. Embora a utilização de painéis centralizados venha a cobrir esta necessidade, muitas vezes a sala de controle possui grandes extensões com centenas ou milhares de instrumentos tornado o trabalho do operador uma verdadeira maratona.


Outra vantagem é que o Software de Supervisão e Controle armazena todas as informações recebidas permitindo ao usuário a análise dos acontecimentos.
Correção de Desvio
Otimização de Processos
Documentação de Partida / Lotes
Isto significa
Mais segurança operacional
Melhor qualidade
Menor custo operacional

O sistema supervisório veio para reduzir a dimensão dos painéis e melhorar o performance homem/máquina.

Baseados em computadores ou microcomputadores executando softwares específicos de supervisão de processo industrial o sistema supervisório tornou-se a grande vedete da década de 80.

O INTOUCH é um software destinado a promover a interface homem/máquina, onde proporciona uma supervisão plena de seu processo através de telas devidamente configuradas.

O INTOUCH possui telas que representam o processo , onde estas podem ser animadas em função das informações recebidas pelo CLP, controlador, etc. Por exemplo: no acionamento de uma bomba, a representação na tela mudará de cor informando que está ligada, um determinado nível varia no campo, a representação na tela mudará de altura informando a alteração de nível. O que o INTOUCH faz é ler e escrever na memória do CLP ou controlador para a atualização das telas.

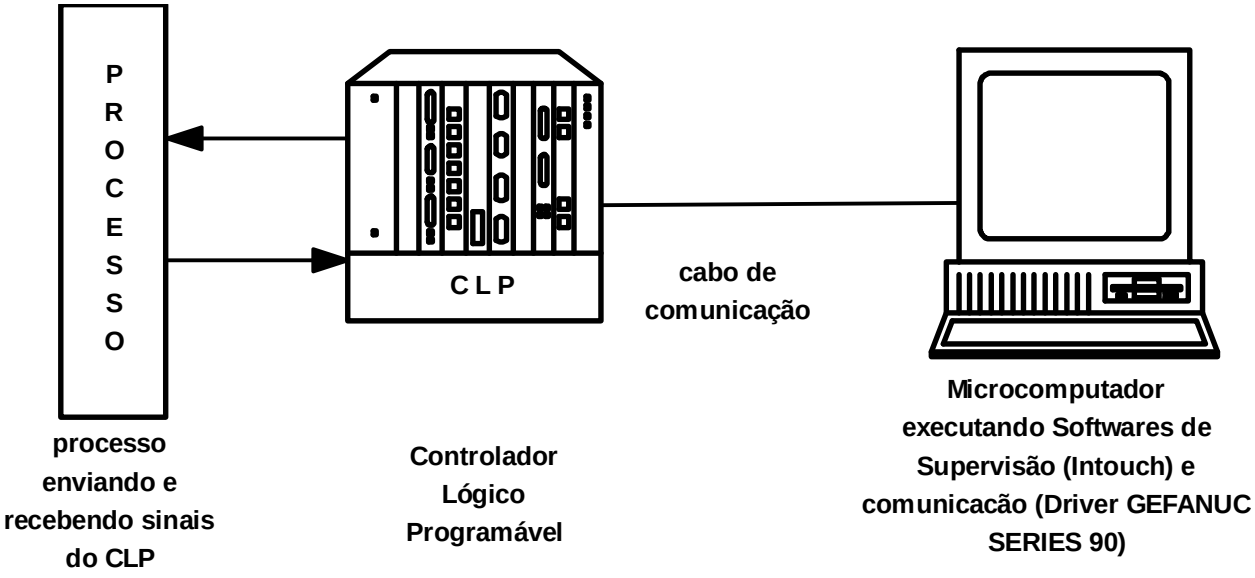
O operador supervisiona e controla todo o processo através de um conjunto de telas que, dentro de um padrão, serão detalhadas de forma especifica para cada processo e indústria.
Hierárquicas Geral
Grupo
Sintonia
Gráficas
Tendências

Quando falamos de supervisão temos a idéia de dirigir, orientar ou inspecionar em plano superior. Através do sistema supervisório é possível de ligar ou desligar bombas, abrir ou fechar válvulas, ou seja, escrever na memória do CLP.

Para a comunicação entre INTOUCH e CLP necessitamos de:

- Hardware : é utilizada uma via de comunicação, que pode ser uma porta serial, uma placa de rede, etc;
- Software : **para comunicação é necessário que o driver do equipamento esteja sendo executado simultaneamente com o INTOUCH.**

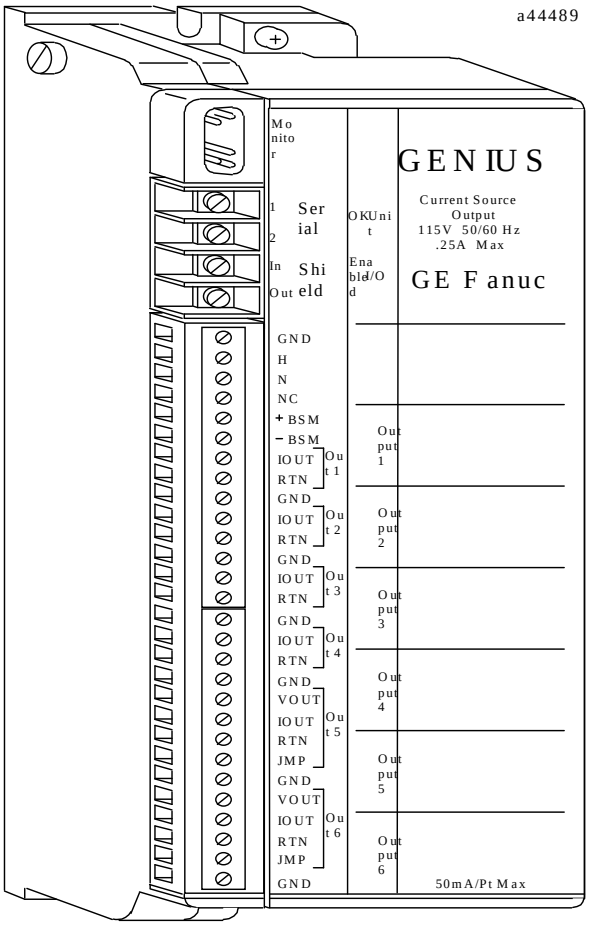
O driver é um software responsável pela comunicação, ele possui o protocolo de comunicação do equipamento.



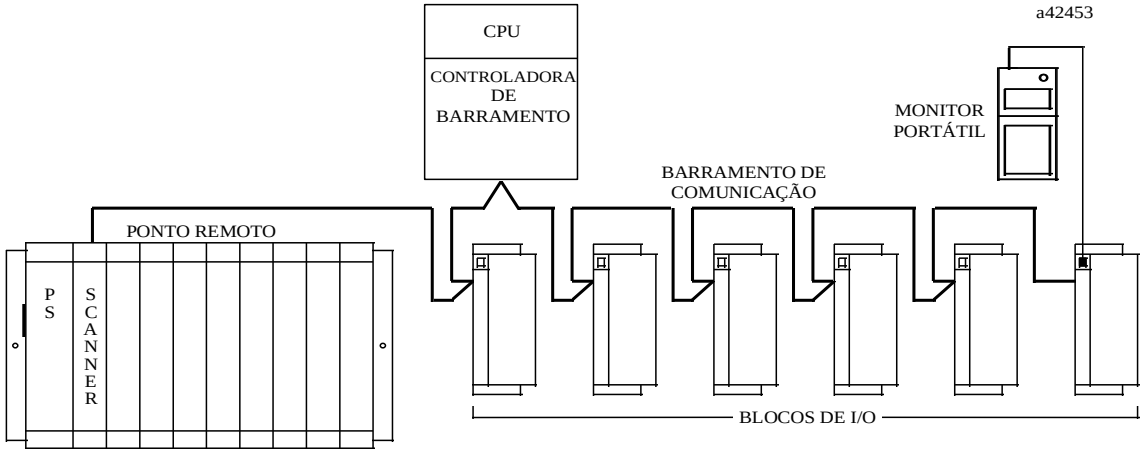
IX. Noções de Blocos I/O Remotos

A instalação de um sistema automático com o uso de I/O locais, requer um gasto considerável de cabeamento, borneiras, caixas de passagem, bandejas, projeto e mão-de-obra para a instalação. Os blocos I/O remotos possibilitam uma redução drástica destes gastos, uma vez que todos os sinais não serão encaminhados para o rack do CLP e sim para pontos de entradas e saídas que ficarão localizados no campo.

Este módulos de I/O são inteligentes, independentes e configuráveis. Interligados entre si através de um barramento de campo, e este a um controlador de barramento que fica localizado no rack do CLP.



A seguir temos a exemplicação da ligação dos blocos I/O remotos.



Um barramento pode atender a:

- **Blocos I/O**, que fornecem uma interface para uma grande variedade de dispositivos discretos, analógicos e para fins especiais. Os blocos são módulos independentes com recursos avançados de diagnóstico e muitos recursos configuráveis por software.
- **Pontos Remotos**, racks de I/O cuja interface com o barramento é feita através de Módulos de Scanner de I/O Remotos. Cada ponto remoto pode incluir qualquer combinação de módulos discretos e analógicos de I/O.
- **Monitor Portátil**, que pode ser usado como um dispositivo portátil ou montado de maneira permanente. Um HHM fornece uma conveniente interface de operador para a configuração de blocos, monitoração de dados e diagnóstico.

Um barramento permite aprimorar o controle de I/O através do uso de comandos de comunicação no programa. O barramento também pode ser usado inteiramente para o controle de I/O, com múltiplos dispositivos de I/O e sem comunicação adicional. Pode ainda ser dedicado à comunicação da CPU, com múltiplas CPUs e sem dispositivos de I/O. Sistemas mais complexos também podem ser desenvolvidos, com CPUs duplas e uma ou mais CPUs adicionais para a monitoração de dados.

Este material didático foi preparado por William da Silva Vianna, professor do curso Pós-Técnico em Automação, curso Técnico em Instrumentação e |Tecnólogo em Automação.
O autor ser reserva no direito de permitir a cópia total ou parcial deste documento, desde que seja citada a fonte. Pegar prontinho e colocar o nome é feio.